NUI Galway
OÉ Gaillimh

ARAN - Access to Research at NUI Galway

Provided by the author(s) and NUI Galway in accordance with publisher policies. Please cite the published version when available.

Title	Graph topology and the evolution of cooperation
Author(s)	Li, Menglin
Publication Date	2015-01-31
Item record	http://hdl.handle.net/10379/4969

Downloaded 2015-12-18T13:34:02Z

Some rights reserved. For more information, please see the item record link above.

Graph Topology and the Evolution of Cooperation

Menglin Li

National University of Ireland, Galway

This dissertation is submitted for the degree of

Doctor of Philosophy

College of Engineering

2014

I would like to dedicate this thesis to my loving parents ...

Acknowledgements

I would like to take this opportunity to thank all of the people who have helped me with the research and thesis.

This thesis would not have been possible without my supervisor Dr. Colm O'Riordan's help. He is a great person that helps me so much on both my research and life during my PhD. I would like to show my gratitude to him.

I would also like to thank for the support of the Irish Research Council for funding under the IRCSET postgraduate scheme.

Lastly, I'll thank my parents and family and everyone in the Computational Intelligence Research Group (CIRG) for their support and honesty, and all the lectures and students of NUIG who have helped me so much in my research.

Abstract

It has been shown that the graph topology can influence the level of cooperation in evolutionary games. Previous research showed that graphs like scale-free networks and small-world networks are more useful in maintaining cooperation. Many researchers believe that the emergence of a high level of cooperation on scale-free networks is due to its power-law degree distribution and the existence of hubs. The main motivation of this work is to examine the influence of the structure of the graph in maintaining cooperation. Apart from the graph topology, we have also examined the influence of the game's payoff and the player's strategies on the emergence of cooperation.

We analysed the payoff matrix of the two-player pure-strategy social dilemma on a lattice grid first, followed by a study of iterated strategies in the prisoner's dilemma game on a lattice grid, from which, we have understood the influence of the game's payoff and the player's strategy. The analysis of the game's payoffs and the player's strategy helped us to select suitable values of those attributes during the subsequent research on complex graphs, through which, we could understand the influence of the graph topology more clearly, and identify more features that influence the outcome of the evolutionary games.

With selected payoff matrices and strategy sets and have run experiments on graphs generated by our new graph generation algorithms; we analysed the influence of the average degree and transitivity of the graphs on the robustness of the cooperation. The results of these experiments show that the robustness of cooperation is actually related to the presence of certain combinations of sub-communities. This finding lead us to the research on individual nodes in graphs. By analysing the graph centralities of each individual node, we found that the graph centrality is related to the ability of certain strategies to invade the population. This gives us a prospective vision that predictions can be made to understand the invasion of the defectors through examining the node centralities. This may help us to predict the level of cooperation on complex graphs.

The unique contributions of this work include:

- The understanding of the influence of game's payoff on the emergence of cooperation

and a prediction formula that can predict the level of cooperation on lattice grids.

- An analysis of the performance of N-player TFT strategies against the pure strategy players (pure C and pure D): N-player TFT strategies can not be guaranteed to outperform pure D strategies, but the N-player TFT strategies with a lower level of tolerance usually perform much better than others.

- A new approaches to generating graphs with predefined degree and clustering coefficient.

- An analysis of the influence of average degree and transitivity of graphs on the robustness of cooperation: in lower average degree graphs, higher transitivity means increased robustness of cooperation.

- Research of how an individual node's graph centrality can affect the level of cooperation, which gives us an approach to making prediction of the level of cooperation through analysing the centralities of the nodes with a defector, in order to helps us to understand the invasion of defection on complex graphs. This could be an alternative approach to understand the evolution on complex graphs.

- Development of an open source graph simulator. This simulator can be found on the website of our research group: CIRG of NUI, Galway.

Contents

Contents v

List of Figures ix

List of Tables xii

Nomenclature xii

1 Introduction 1

1.1 Introduction . . . 1
1.2 Game Theory and Evolutionary Computation . . . 1
1.3 Motivation and Objectives . . . 3
1.3.1 Hypotheses . . . 4
1.3.2 Methodology . . . 6
1.4 Expected Outcome and Potential Contribution . . . 6
1.5 Thesis Layout . . . 7

2 Research Background 9

2.1 Introduction . . . 9
2.2 Evolutionary Computation and The Theory of Games . . . 9
2.2.1 Evolutionary Computation . . . 9
2.2.2 Introduction to The Game Theory . . . 10
2.3 Evolutionary Game Theory . . . 15
2.3.1 Introduction to Evolutionary Game Theory . . . 15
2.3.2 The Evolutionary Stability and The Population Dynamics . . . 17
2.4 Introduction to Graph Theory . . . 18
2.4.1 Topology of The Graphs . . . 19
2.4.2 Communities . . . 20

2.4.3 Graph Centralities and Properties . . . 22
2.4.4 Graph Generation Models . . . 30
2.5 Evolution on Graphs . . . 33
2.5.1 Graph Topology Effects the Evolution of Cooperation . . . 33
2.5.2 The Evolution of the Graph Structure . . . 37
2.6 Summary . . . 37

3 Evolutionary Games on a Lattice Grid 39
3.1 Introduction . . . 39
3.2 Payoff Matrix and Payoff Plot . . . 39
3.2.1 Linear pure strategy games . . . 39
3.2.2 Non-linear game and mixed strategy . . . 43
3.2.3 Normalization . . . 46
3.2.4 The invasion of strategies . . . 49
3.3 Experiments . . . 53
3.3.1 The prediction . . . 54
3.3.2 Experiment results . . . 55
3.3.3 The definition of different games . . . 58
3.4 Summary . . . 60

4 The Iterated Strategies 62
4.1 Iterated Games . . . 62
4.2 N player prisoner dilemmas . . . 62
4.3 Experiments and results . . . 64
4.3.1 Iterated games on random initialized lattice graphs . . . 64
4.3.2 Iterated games on pre-set patterns . . . 70
4.4 Summary . . . 72

5 Graph Topology and The Robustness of Cooperation 74
5.1 From Regular to Complex Graph . . . 74
5.2 Generation of graphs with adjustable clustering coefficient . . . 75
5.2.1 Ascending graph generate algorithm (ascGen) . . . 77
5.2.2 Heuristic graph generation algorithm . . . 78
5.3 Graph property and the emergence of cooperation . . . 78
5.3.1 Graph architecture . . . 78
5.3.2 The robustness of cooperation . . . 81

5.4 Experimental result 84
5.4.1 Cooperation on graphs with different clustering coefficients without considering the average degree 84
5.4.2 Cooperation on graphs with different clustering coefficients taking into account the average degree 85
5.5 Summary 86

6 Graph Centralities and the Invasion of the Defection 88
6.1 From Community to Individual 88
6.2 Robustness of graph and robustness of individual 89
6.2.1 Cooperation of the graph and the invasion of individuals 89
6.2.2 Invasion of a defector 90
6.2.3 Centrality measures that may influence the spread of defection . . . 91
6.3 Experiment result 92
6.4 Summary 99

7 Conclusion and Further Work 101
7.1 Conclusion 101
7.2 Future Work 104

References 106

Appendix A The Graph Simulator : A User Manual 115
A.1 Introduction 115
A.2 System Requirements 115
A.2.1 Running the Simulator 115
A.2.2 Compiling the Source Code 116
A.3 User Manual 117
A.3.1 The Main Scene 117
A.3.2 The Graph Control Toolbar 117
A.3.3 The Game Control Tool-bar 121
A.3.4 File Format 123
A.3.5 Create and Browse Graphs Using the Graph Browser 125
A.3.6 Setting up the Games 127
A.3.7 Using the LUA Console to Scripting 127
A.4 HotKey List 130

A.5 summary 131

Appendix B Publications 132

List of Figures

2.1 Distribution of social dilemma games in the S-T plot 13
2.2 Adjacency Matrix 19
2.3 Example of Complex Graphs 20
2.4 Example of Degree Centrality 23
2.5 Example of Clustering Coefficient Centrality 25
2.6 Example of Closeness Centrality 26
2.7 Example of Betweenness Centrality 28
2.8 Example of Eigenvector Centrality 30
2.9 Nowak and May's visualisation 35

3.1 General payoff plot of a two strategy game 40
3.2 Nash equilibrium payoff plot 42
3.3 Non-linear game's payoff function 44
3.4 Mix strategy 45
3.5 Normalized payoff plot 48
3.6 An example of the invasion in the Moore neighbourhood. P_0 is the current player, the cooperator with the highest payoff is P_C and the defector with the highest payoff is P_D. Realise that both P_C and P_D are in P_0's neighbourhood NH_0, but P_C and P_D do not have to be in the same neighbourhood. The minimal number of common neighbour of P_C and P_D is one, which is P_0. . . 50
3.7 The emergence of different patterns over different value of β and α in the pure strategy, 2-player prisoner's dilemma game. Each line represents the prediction function with different values for m and n. 56
3.8 For example, when the value of α and β is undertaken in the green invasion level, the evolution is highly unstable; the cooperation rate over generation is plotted. 59

4.1 Game with pure C/D strategy set, pure C/D, TFT4 strategy set, and pure D and TFT4 strategy set start with random initialization $\beta = 1.51$ 66
4.2 The TFTN strategy player playing with Pure C/D strategy players start with random initialization $\beta = 1.51$ 67
4.3 The TFTN strategy player playing only with Pure D strategy players start with random initialization $\beta = 1.51$ 67
4.4 Game with pure C/D strategy set, pure C/D, TFT4 strategy set, and pure D and TFT4 strategy set start with random initialization $\beta = 1.61$ 68
4.5 The TFTN strategy player playing with Pure C/D strategy players start with random initialization $\beta = 1.61$ 69
4.6 The TFTN strategy player playing only with Pure D strategy players start with random initialization $\beta = 1.61$ 69
4.7 Pure C/D and TFTN strategy randomly mixed, $\beta = 1.51$ / $\beta = 1.61$ 71
4.8 Pure C/D and TFTN strategy mixed, has $1/3$ pure D at the first generation. $\beta = 1.51$ / $\beta = 1.61$ 71
4.9 One defector's invasion in a group of TFT0 (pure C) to TFT8 players $\beta = 1.61$ 72

5.1 The graph created by 6 complete sub graphs of size 5 80
5.2 The defectors invade the entire graph in a high degree graph 82
5.3 One defector in the sub graph; as the vertex on the bridge linking to the other sub graph will get a smaller payoff than the cooperator in the other sub graph that is connected to it; it will continue swapping its strategy between cooperate and defect, and never invade other sub graphs. 83
5.4 The defector on the bridge may invade two sub graphs. 83
5.5 Without controlling the number of edges, the experiment result is not significant enough to show the influence of the clustering coefficient on the robustness of cooperation. 85
5.6 With the same size, the cooperation rate of the graph is linearly increasing as the clustering coefficient increasing. 86

6.1 The cooperation rate of each individual in graph with transitivity value 0.5 . 93
6.2 The average number of individuals that can not spread defection in the graph with specific values of transitivity. 94

6.3 Cooperation rates after 100 generations for the graphs with different transitivity and average degree. The number of edges is specified above the graphs; the colour of the lines defines the transitivity; the x-axis plots the index of the node that have been set to defect; the cooperation rate is plotted on the y-axis. 95
6.4 Individual centrality by the order of each individual's spread of defection in graph 5.1, part 1 96
6.5 Individual centrality by the order of each individual's spread of defection in graph 5.1, part 2 97
6.6 Individual centrality by the order of each individual's spread of defection in graph 5.1, part 3 98

A.1 The Graph Player main interface 117
A.2 The Graph Control Bar 117
A.3 Graph displayed using ring layout 119
A.4 Graph displayed using the Spiral layout 119
A.5 Graph displayed using the Square layout 120
A.6 The Game Control Bar 121
A.7 Sample Graph 123
A.8 Sample of the Graph Properties 125
A.9 Sample of Multi-Selection 126
A.10 Sample of displaying all neighbours for a node 127
A.11 Opening the Lua Console 128
A.12 Using Lua function to get and set the temptation payoff 128
A.13 Using Lua functions to add Nodes and Edges 129
A.14 More lua scripts in console. 130

List of Tables

2.1 Payoff table for the row player in 2 player, N strategy games 11

3.1 Payoff of a two-player game 40

A.1 V/E file format sample 124
A.2 Adjacency matrix file format sample 124

Chapter 1

Introduction

1.1 Introduction

In our daily life, we have to make decisions. A good decision may benefit us a lot in the future, and a bad decision may make us suffer from a bad result. How to make a good decision has always been an interesting topic, and many approaches have been proposed since the beginning of human culture. The theory of "how to make decisions" is studied in "game theory", and the "approaches to make decisions" are called strategies. Since modern game theory has proven successful in analysing individual behaviour, it has been quickly adopted into many domains such as economics. Despite the success of it in many domains, many questions remain unanswered in the analysis of living organisms.

Considering the evolution and the learning mechanisms of living creatures, "evolutionary game theory" have been introduced to help understand the behaviours of human and animal behaviours by biologists. In the original games, all players were assumed to be fully rational, and know the payoff of other players. However, this can not be assumed in many scenarios in the real-world. Evolutionary games, instead, allow each player to have individual behaviours, and to learn and to evolve new strategies over generations. This will allow evolutionary game theory to be able to simulate some natural processes more precisely.

1.2 Game Theory and Evolutionary Computation

Research in modern Game Theory began with John Von Neumann in the early 20th century [123], and increased in popularity after John Nash developed concepts such as the Nash equilibrium to study player's strategies in the 1950s [75]. Since then, game theory has been

widely adopted in many different domains. The original game theory was established on one-round games played by two fully rational players. Applying game theory in biology, John Maynard Smith proposed to play the games iteratively over a large group of players, where each player is not assumed to be "fully rational" [68]. This approach has been further developed into evolutionary games wherein successive generations of strategies are evolved based on their performance in games [40, 128].

Traditionally, mathematical analysis of interactions between strategies represented the dominant approach in game theory. More recently, given the complexity of scenarios under investigation, computational simulations have become the more widespread approach. These simulations have been used to explore many avenues—changing game payoffs [27, 37, 81], changing strategies of players [8, 59, 60], effects of spatial constraints on agent interactions and many others [53, 61, 71, 86, 89, 111, 125]. Several scenarios that have been considered in simulation based approaches include:

1. Exploring the outcomes for different games: by changing the parameters of the game, positive or negative effects on the emergence of cooperation may be witnessed in an evolutionary setting. Given similar spatial organisations, different games such as the prisoner's dilemma and the snowdrift game may lead to different outcomes [37, 66].

2. Exploring the effect of strategy sets and of different learning approaches: researchers have also looked at the effect of an individual's strategies [60, 61] or an individual's learning mechanisms [27, 122]. Adopting certain learning mechanisms or allowing certain types of strategies can also promote cooperation.

3. Exploring the effect of the representation of agents has on the emergence of cooperation in standard simulations but also in simulations where agents are allowed to learn [3, 8].

4. Different spatial constraints represented by different graph topologies can affect the evolution of cooperation. It has been shown for example that clusters of co-operators can be robust to invasion by non-cooperative strategies in certain settings. The investigation into topologies and settings of parameters defining the graph is an ongoing topic of investigation particularly in small world and scale free graphs [2, 79–82, 94, 95, 104, 106, 124].

Apart from the mathematical interest, research into evolutionary games has also been widely adopted in many domains, due to the advantage of saving both time and cost using

evolutionary game simulations. In comparison to collecting real world data and / or doing experiments using human participants, the research in evolutionary game simulations is not only popular among computer scientists, mathematicians and physicians, but also popular in many other domains including biology / biomedicine [1, 20, 49, 83, 121], and social science [9, 38, 99].

1.3 Motivation and Objectives

It has been proposed that mutual cooperation plays a significant role in nature [78]. Although there is evidence that shows defection is often preferred in certain naturally occurring scenarios, cooperation still widely exists in many scenarios. Since mutual cooperation can benefit the entire society, unlike defection which can only gain the maximum payoff for the defector, high levels of cooperation is widely desired in many situation and domains. It is commonly believed there are certain constraints that may be of benefit to the emergence of cooperation, and make cooperators more robust against the invasion of defectors. Experiments have shown that the payoff functions, learning mechanisms, and the network structures are all have significant influence on the cooperation on complex graphs.

In this research, the author is trying to explore the mechanisms that may influence the levels of mutual cooperation during the evolution, under different constraints of the game strategy / pay-off and the graph topology. The research will addresses linear two player social dilemmas (such as prisoner's dilemma, snow drift game, stag hunt game, etc.), with some strategies such as pure C (always cooperate), pure D (always defect), and a set of N-player TFT (Tit for Tat, defect when others defect) strategies, covering a wide range of graphs from lattice grid to complex network with several tunable parameters. The aim of the research is to explore the potential settings of the games or the specific network structure, which may have positive influence on the emergence and robustness of the cooperation from the aspect of both the entire population and the individual players.

The objectives of this research include:

1. Quantify the different influences on the emergence and robustness of cooperation in different social dilemma games (or the games with different payoff matrices), analyse and attempt to capture the correlation between the game payoff values and the cooperation rate at the evolutionary stable state on regular graphs (such as lattice grids).

2. Incorporate iterated strategies such as Tit for Tat strategy (TFT) into N-player social dilemma games and explore if the iterated strategy (which have been shown to in-

duce cooperation in two player iterated games) can dominate the population in spatial population.

3. Design and run experiments on graphs with different attributes, observe the robustness of the different structured cooperative populations under the invasion of a defector, and identify the communities which most induce cooperation.

4. Explore the correlation between an individual's centrality in the graph and the spread of the defection from this individual in a fully cooperative population; attempt to a measure an individual's robustness will respect to cooperation.

Overall, this research is looking for potential patterns or certain community structures in an individual's neighbourhood that can provide high robustness which respect to cooperation, or can promote the emergence of the cooperation to the entire graph / society. Such structures could be of benefit to many public research such as information retrieval, data mining, community detection, cognitive science, virus control, advertising, evolutionary biology and biomedicine, etc.

1.3.1 Hypotheses

This research will consider both the payoff matrix of the games, and the evolutionary dynamics of the population. The main goal of the study is to discover the correlation between the graph topology and the emergence and robustness of the cooperation. The research will start with two player social dilemma games on a lattice grid, and end with complex graphs with multiple adjustable attributes. The research will also include the analysis of the invasion of defections based on the game's payoff matrix, and the iterated strategy such as N-player Tit for Tat (TFT). Several new algorithms / measures are introduced during the research, such as the algorithms to generate graphs with pre-defined clustering coefficient, and the clustering eigenvector centrality measure.

By considering the co-evolution on the lattice grid which has been first introduced by Nowak and May in their paper [81], we studied the evolution over the spatial structure, in particularly, the Moore-Neighbourhood where each player interacts with its 8 immediate neighbours. By analysing the correlation between the game's payoff matrix and the evolutionary patterns, we hope to develop a prediction function for the potential cooperative patterns on lattice grid, which demonstrates the invasion of a single defector in a fully cooperative population, in the spatial structure.

We also adopted a series of iterated strategies in the games, and explored if the iterated strategy would be preferred among the players in the spatial population. The iterated strategy has been shown to win most games against pure strategy players in the 2 player games [5, 6]. However, it may or may not be favoured in a complex network. The result of the iterated strategies on a lattice grid will be considered by us to decide whether we'll adopt iterated strategies in the complex network.

After examining the influence of the game's payoff matrix and the strategy on the evolution, we'll decide to adopt a particular game, such as a 2-player prisoner's dilemma game with a payoff matrix $\begin{pmatrix} 3 & 0 \\ 5 & 1 \end{pmatrix}$, and then normalise it to following format: $\begin{pmatrix} 1 & 0 \\ \beta & 0 \end{pmatrix}$, with fixed values of β in the graph topology experiments. This will decrease the influence of the game's setting to the minimum and emphasise the effect of the graph topology.

Two new graph generation algorithms are developed to generate graphs with certain desirable attributes. A simulator has been developed to run the experiments as well as to measure and monitor the graph centralities during the evolution. From both the "global" and "individual" views, we can examine the correlation between the graph's attributes, individual centrality, and the robustness of the cooperation.

According to the points above, we have the following hypotheses for this research:

H1 There exists a correlation between the game's payoff matrix and the emergence of cooperation in the lattice grid, and this correlation can be used to generalise a prediction function. The invasion of the defection in regular graphs can be calculated from this prediction function.

H2 The iterated strategies in spatial graphs have different performance than the pure strategies.

H3 New graph generation algorithms can be developed to generate particular graphs with pre-defined value for the number of vertex, number of edges, and the transitivity.

H4 Graphs with different attributes performs differently with respect to the emergence and robustness of cooperation.

H5 There exists a correlation between centrality values of individual nodes and the spread of defection from each individuals in the graph.

1.3.2 Methodology

The main method of this research will be through simulations. Following from these hypotheses, we'll review the corresponding literature for each part of the research, and experiments will be designed. Following this, simulators and tools (such as the graph generator) will be built in order to complete each experiment. Several versions of the simulator are built to allow experimentation throughout the process. Analysis of the evolutionary dynamics and outcomes in the simulations will be central to the research process.

The entire research will be broken into to four main steps, including:

1. The research into the effect a game's payoff may have on the evolution and cooperation on lattice grid.
2. The research of N-player iterated strategies in spatial games.
3. The research into population structure and its effect on the emergence of cooperation.
4. The research of graph centrality and the spread of defection from individual nodes.

In each section of the research, we'll review the related literature, design and set up experiments, and develop the necessary software tools for the experiments. Mathematical analysis and result analysis will also be shown in each section.

Furthermore, in each section, we will provide information from previous research, from the analysis and the experiment results to motivate to the next step.

The graph that is being researched in each step will vary from a regular graph (lattice grid) in the first two steps, to a complex graph with multiple adjustable parameters in the last two steps.

1.4 Expected Outcome and Potential Contribution

As cooperation has been proven to be a key role in natural evolution [78] and many other domains [49, 99], much research has been undertaken to examine cooperation from different aspects. This research makes contributions to the study of the correlation between the graph topology and the emergence / robustness of the cooperation. The theoretical underpinning of this research is the "five rules for the Evolution of Cooperation" introduced by Nowak in his paper [78], which believes that "Kin Selection", "Direct Reciprocity", "Indirect Reciprocity", "Network Reciprocity" and "Group Selection" can be critical to cooperation. Although much research have been done in this domain, there are still open questions waiting to be solved.

The expected outcomes from this research include:

1. A prediction function of the evolutionary patterns on lattice grids with Moore-Neighbourhoods.
2. A comparison of the iterated strategies with pure strategies on spatial graphs.
3. A definition of how graph attributes, especially average degree and transitivity can affect the robustness of cooperation of a graph.
4. An explanation into the correlation between an individual's graph centrality and the spread of defection from this individual in a cooperative population.

The results of this study may have contributions to the research in many domains; examples include: we provide a potential useful measure to add to recent research on "Game centrality", which shows that "Game centrality" (which is the spread of defection in our thesis) can be a important measure in many real world domains [112].

1.5 Thesis Layout

The structure of this dissertation is as follows:

- Chapter 1 introduces evolutionary game theory, and explains our motivation, research methods, hypotheses, and the expected outcomes and contributions.
- Chapter 2 reviews the literature of the related work, introduces the research background, and discusses the open questions in the domain.
- Chapter 3 presents research on the social dilemma games on the spatial graphs.
- Chapter 4 introduces the N-player Tit For Tat (TFT) strategies, and compares it with pure strategies.
- Chapter 5 defines two new graph generation algorithms based on the configuration model, which can generate graphs with tunable clustering coefficients. Experiments will be run to examine the influence of average degree and transitivity (average clustering coefficient) on the evolution.
- Chapter 6 will present further study on the spread of defection from individual, graph centralities such as degree, clustering coefficient, betweenness, closeness, eigenvector, etc. will be examined based on the node's ability to spread the defection.

- Chapter 7 presents the final conclusions, discusses our contribution, and the potential for future work.

Chapter 2

Research Background

2.1 Introduction

The research in this thesis is focussed on both game theory and graph theory. In this chapter, we'll introduce the concepts that we use in later chapters, as well as the findings in previous research. We'll discuss game theory and evolutionary games first, followed with the introduction to graph topology, centrality, and communities. We also present some of the previous graph generation models in this chapter, in order to compare to the new generation algorithm that we'll introduce in the later chapter.

2.2 Evolutionary Computation and The Theory of Games

2.2.1 Evolutionary Computation

The idea of evolutionary computation is to use computational approaches to solve real-world problems over generations in a population based system. Inspired by the evolution of living organisms, earlier researchers developed different approaches to solve complex problems, such as evolutionary programming [32], and genetic algorithms [50]. Such methods are referred to as evolutionary computation. The general form of evolutionary algorithms usually uses a large population, updating the population from generation to generation with operators such as mutation, crossover etc. In the last decades, the success of evolutionary computation have been shown in many domains, especially in the problems which have a complex solution space, and are hard to model mathematically (normally referred to as "NP-Complete" problems).

Being one of the most important research domains in computational intelligence, the

research of evolutionary computation can be categorized into many sub domains including genetic algorithms, genetic programming, and grammatical evolution. Currently, the application of evolutionary computation have been widely adopted in many domains such as information retrieval, data mining, biological science and cognitive sciences.

Moreover, many current research methods have been inspired / influenced by evolutionary computation. The concept of "iteratively learning from others" has been adopted in many different algorithms and domains as a method to study certain properties. For example: in game theory, games have been developed to learn and adopt strategies form other more successful players.

2.2.2 Introduction to The Game Theory

As an approach to study the individual strategies in decision-making, the research of modern game theory can be traced to the 19th century, to the book *Theory of games and economic behaviour*, by John Von Newman and Oskar Morgenstern in 1944 [123]. The original game theory is designed to study the decision making for individuals under circumstances where every player is assumed to be fully rational. The study in game theory became popular after the well-known concept, *Nash equilibrium*, had been introduced in 1950, by John Nash [75]. Nash equilibrium among strategies refers to the scenario where no strategy can unilaterally deviate and obtain a better payoff. This has been used as the fundamental theory in the establishment of the modern economic theory.

There are many factors that may influence the decision of players. One of the most important attributes is the payoff (the outcome of the player's decision) of the strategy (the player's decision). Different types of payoffs can be viewed as different games. There are many games that have been introduced so far such as the prisoner's dilemma, the public goods game, etc. Games that have similar features can be categorized into one group, and studied together, such as the social dilemma games.

The Social Dilemma Games

Social dilemma games are one of the most common type of games that have been researched. A social dilemma game allows players to choose either to get the maximum payoff for themselves (selfish behaviour) or to contribute to the gain for the whole society. Social dilemma games capture the conflict between choosing individually rational actions and actions that are for the collective good. Examples include the *Prisoner's Dilemma Game*, *Snow Drift (or hawk-dove) Game*, *Stag Hunt Game*, and the *Public Goods game*, etc. The strategies of

	Strategy 1	Strategy 2	...	Strategy M
Strategy 1	Payoff 1_1	Payoff 2_1	...	Payoff M_1
Strategy 2	Payoff 1_2	Payoff 2_2	...	Payoff M_2
...	...	...	...	...
Strategy N	Payoff 1_N	Payoff 2_N	...	Payoff M_N

Table 2.1 Payoff table for the row player in 2 player, N strategy games

social dilemma games are often viewed as "cooperation" and "defection". The players will receive a "reward payoff" for mutual "cooperation", but it can not reach the personal highest payoff without playing "defect" against other "cooperators". By adopting the "defection" strategy, a player may receive its personal highest payoff if the opponent plays "cooperate"; this payoff is usually referred to as the "temptation payoff"; however, this will damage all the other players' payoffs in the same society. Finally, mutual "defection" will lead to a "punishment payoff" for both the players.

A payoff matrix is usually used to represent the payoffs that a player will receive for adopting a certain strategy. For example, for a two-player, N-strategy game, the payoff of each strategy for the row player have been showed in the following Table 2.1.

Payoff tables like above usually can be represented as an $N \times N$ payoff matrix. Different games may have different payoff functions, a general form of the payoff matrix for any two-player social dilemma games can be present as:

$$P = \begin{array}{c} \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cc} C & D \end{array} \\ \begin{pmatrix} R & S \\ T & P \end{pmatrix} \end{array}$$

In the payoff matrix above, C and D stand for the "Cooperate" and "Defect" strategies for the players respectively. R is the "Reward" payoff for being mutual cooperative; S is the "Sucker" payoff for a cooperator playing against a defector; T is the "Temptation" payoff for a defector playing against a cooperator, and P is the "Punishment" payoff for mutual defection. A social dilemma usually satisfies the following conditions: $T >= R >= max(P,S)$. A second constraint, $2R >= T + P$, is usually added for iterated games.

The constraints across different social dilemma games also differ slightly. A prisoner's dilemma game (PD) usually needs to satisfy the condition: $P >= S$ in addition to the above conditions, while a snow drift game (SD) usually satisfies: $S >= P$ [55]. The two-player stag-hunt game (SH) is similar to the PD game but the reward payoff is greater than the

temptation payoff; ie: $R > T \geq P > S$.

To decrease the number of variable parameters in the payoff matrix of the social dilemma games, normalisation is often applied to the raw payoff matrix. A common approach to normalise the social dilemma is to scale the rewards such that the reward is set to 1 and the punishment payoff is set to 0 [93, 108, 116].

After the normalisation (a basic matrix transformation by first adding $-p$ and them multiply $1/R$ to all entries of the matrix), the payoff matrix can be rewritten to:

$$P = \begin{pmatrix} 1 & S \\ T & 0 \end{pmatrix}$$

where $-1 \leq S \leq 1, 0 \leq T \leq 2$.

If we set the constraints to be $0 \leq S \leq 1$ and $0 \leq T \leq 1$, the Nash equilibrium strategy is to cooperate. This game is known as the harmony game and is not a social dilemma game.

Adopting the normalisation approach described above, the payoff matrix of any 2-player prisoner's dilemma can be represented as follows:

$$P_{2PD} = \begin{pmatrix} 1 & 0 \\ \beta & 0 \end{pmatrix}$$

where the only variable in the normalised payoff matrix β is the normalised temptation payoff.

The payoff matrix of the snow-drift game can be represented as:

$$P_{SD} = \begin{pmatrix} 1 & 1-\beta \\ 1+\beta & 0 \end{pmatrix}$$

The payoff matrix of the stag-hunt game can be represented as:

$$P_{SH} = \begin{pmatrix} 1 & -\beta \\ \beta & 0 \end{pmatrix}$$

The games can be viewed graphically with respect to T and S as depicted in Fig.2.1

From Figure 2.1, we can see that except for the harmony games, all of the 2-player social dilemma games can be categorized in to one of the three social dilemma games: Prisoner's dilemma game, Snow Drift game, and the Stag Hunt game.

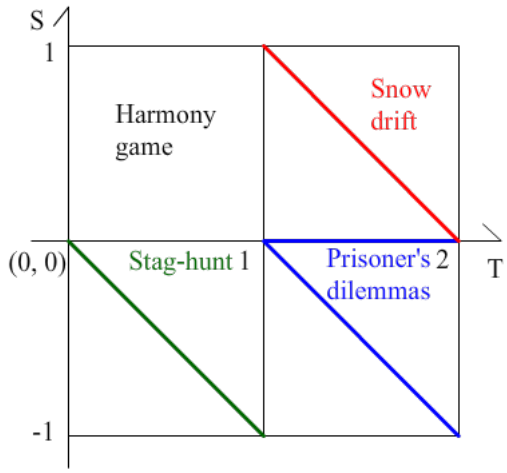

Fig. 2.1 Distribution of social dilemma games in the S-T plot

Strategies of the Game

In game theory, the term strategy refers to decisions that a player can make in a game. A player can decide whether to cooperate with other players or when to cooperate or defect, those decisions are the strategies that the player can take. Adopting different strategies can directly influence the payoff that the player receives from the game. Individual strategies that players can only adopt are referred as to pure strategies. We have already introduced two pure strategies in social dilemma games in the last section, which are the cooperation strategy and the defection strategy.

Other than the pure strategies (pure C and pure D), there are many other strategies that can be adopted during the games, such as the mixed strategy [76, 123], which allowed players to adopt different strategies in different moves based on probability; quantum strategies [28, 29, 69, 70], which allows players to adopt a super-position of different strategies between moves; and iterated strategies [5, 6], which allows the players to select strategies according to its personal history of moves and outcomes. In game theory, if the strategies of each player are know, and one player can not change its payoff without changing other player's strategies, then there is a Nash equilibrium, equilibrium for short. If every player's strategy is pure strategy, then the equilibrium is a pure strategy equilibrium. There are no guarantee that every game has a pure strategy equilibrium, however, strategies that introduced above may introduce new equilibria in games which do not have pure-strategy Nash equilibria, and makes cooperation more robust, or mat increase the payoff value under

certain circumstances.

- **Pure Strategy**: A pure strategy is the basic decision a player can make in a game. A player's strategy set is the set of pure strategies that the player is able to adopt.
- **Mixed Strategy**: Mixed strategy defines a probability distribution over different decisions; it defines the probability of the player adopting each pure strategy from the strategy set. Unlike pure strategy Nash equilibria, at least one mixed strategy Nash equilibrium exists in each zero-sum game with finite strategy set [76].
- **Quantum Strategy**: A quantum strategy is a strategy that allowed a player in a super position of different strategies. During a quantum game (game that allowed player adopted quantum strategies), when a player change its strategy, it may influence other player's strategy. This is the entanglement of players. Although mixed strategy equilibria exist in all zero-sum finite strategy games, the mixed strategy equilibria can not increase the expected payoff of a player. Examples have been showed that quantum strategies can actually increase the expected payoff in quantum games such as the PQ penny flip game [69, 70]. There are a few mathematical models that have been developed so far, the most famous one is the "EWL" model introduced by Eisert, Wilkens, and Lewenstein [29], which has been used to develop many quantum games such as the quantum prisoner's dilemma and the quantum chicken game [28]. The current research in quantum games mostly uses 2-player games.
- **Iterated Strategy**: The iterated game has been researched more and more extensively following Axelrod's work in the iterated prisoner's dilemma [6]. In his book, Axelrod introduced the Tit For Tat (TFT) strategy, which is defined as always cooperate on the first move, and then copy the opponent's move for subsequent turns. Axelrod showed the robustness of the TFT strategy, and claimed that although TFT may not outperform any of its individual opponents, it may perform well by inducing cooperation among cooperators [6]. Axelrod also showed that the TFT strategy dominated in a tournament competitor similar to evolutionary style games [5, 6]. The majority of work on the iterated prisoner's dilemma has concentrated on the 2 player game although there is also considerable interest in the N-player game [18].

Game Theory Research

Research in game theory began with analysing game rules and strategies. Traditional game theory was introduced by Von Neumann and Nash [76, 123], which concentrated on the

individual strategy in "one-shot" games. There are many games and strategies that have been developed during the last decade, including mixed strategies, iterated strategies and quantum strategies. Although there are many new research topics that have been developed recently in the evolutionary game theory, research on individual strategies is still a "hot topic" in the domain.

Some researchers started to use evolutionary computation in game theory, leading to the development of evolutionary game theory [68, 113]. As the number of studies in the domain of evolutionary computation increased, evolutionary game theory has become more important in the research of game theory.

The research on evolutionary game theory has combined many different domains. Besides game theory, graph theory is also an important research area in evolutionary game theory. Whether or not the graph topology can influence the cooperation has becomes one of the important open questions in current research in evolutionary game theory. Moreover, the way that individuals learn from each other may also make a great impact on cooperation.

2.3 Evolutionary Game Theory

2.3.1 Introduction to Evolutionary Game Theory

Classical game theory typically focused on one-shot games [75, 123]. More recently, rather than analysing the equilibrium of one-shot games, many researchers began to focus their attention on looking for evolutionary stability in evolutionary games which were introduced by John Maynard Smith [68, 113]. The study of evolutionary game theory began in the 1980s. Ideas from evolutionary theory have been adopted into game theory. Having a population of different types of players interacting with each other, Maynard Smith and Price [113] discussed the rate of growth or decline of the different types of players (players that play different strategies) during the evolution. They introduced the concept of the evolutionary stable strategy (ESS); this refers to a strategy which cannot be successfully invaded by any other strategy. The ESS may not exist in some evolutionary games, so, we need another approach to analyse the dynamics of the system. Following the research of Taylor and Jonker [120], the distribution of strategies in the population during the evolution can be calculated using replicator dynamics; replicator dynamic equations describe the rate at which strategies will change over time as a function of the payoff and proportions of all strategy types in the population, and it could be used to describes the dynamics of the system. There are fixed points that exist in the replicator dynamics, which refer to the populations that

are no longer evolving. A relation can be found between the fixed point and replicator dynamics equation and the ESS. The concepts of evolutionary stable strategies and replicator dynamics have been widely studied by many researchers [40, 128].

Research into evolutionary games was originally undertaken for agents located on a complete graph, i.e., panmictic populations where all agents were equally likely to interact with all others. Previous research [40, 68, 113, 120, 128] showed that the replicator dynamics of a dynamic population can be calculated through a replicator equation, and the ESS of the game, if it exists, corresponds to an asymptotically stable fixed point of the replicator dynamic. However, scenarios involving evolutionary games on general graphs (such as the small-world network, the scale-free network, etc.) are more complicated. Given the difficulty in calculating the ESS of the evolutionary games on general graphs, computer simulation has been adopted as a main tool in the research in evolutionary game theory.

To date, the research in evolutionary games includes a vast amount of scenarios. Some of the research explores the evolutionary process and the emergent outcomes while other research concentrates more on the evolution of the strategies themselves [31, 87, 88, 133]. For example, Yao and Darwen have found that increasing the size of the population of players seems to have negative effect on the cooperation rate; they have also ran experiments on different evolutionary environments to test the generalisability of the evolutionary games [133]. Fogel's experiments showed that the payoff matrix is the main factor in deciding the probability of evolving cooperation [31]. O'Riordan presents work exploring the evolution of "forgiving" strategies, and demonstrated that the cooperation could be promoted by the incorporation of forgiveness [88].

Many researchers have attempted to explain and understand the effect that different types of graphs may have on the evolutionary process. Nowak and May found that the use of a spatial topology (a regular lattice) to constrain the agent interactions can benefit cooperation in the evolutionary games [79–82]. More recently a far more extensive range of graph topologies have been explored [102, 130]. Chiong et al. have examined how the structure of the neighbourhood on the lattice grid can affect cooperation in an N-player iterated game [23]. He introduced several neighbourhood structures to run the bidding game and showed that the outcome of the game differs with different types of neighbourhood.

Over all, the research in evolutionary game theory can be categorised into three domains:

1. The research of the game rules and strategies (The parameters of the game, strategies, and equilibrium, ESS, etc.).

2. The learning between individuals (The learning mechanism, strategy exploration, evo-

lutionary dynamic etc.).

3. The structure of population (The effect of neighbourhood structure, cluster coefficient, small world nature of graph etc.).

2.3.2 The Evolutionary Stability and The Population Dynamics

In evolutionary game theory, Evolutionary Stable Strategy (ESS) usually used to refer to a set of strategies adopted by individuals in a population such that no player can benefit by switching to any alternative strategy [68, 113]. The ESS can tell us the stable state of the dynamic system; it can be both pure and mixed strategy.

However, there are no guarantee that an ESS may exist in an evolutionary game. Therefore, the replicator dynamic has been introduced to understand the dynamics of the evolution [120].

Considering the following evolutionary game, each player of the game can adopt a pure strategy from a strategy set $S = s_1, s_2, ..., s_k$. Assuming, the probability of the player learning from other player is α. At time t, the proportion p of the individuals that are playing strategy s_i is defined as $p_i(t)$, $\pi_i(t)$ is the expected payoff of a individual playing strategy s_i, and the average of $\pi_i(t)$ is $\overline{\pi}(t)$, then we get:

Replicator Dynamic Definition 1

$$\frac{d}{dt}p_i(t) = \alpha p_t(t)[\pi_i(t) - \overline{\pi}(t)]$$

The equation above is referred to as the "replicator dynamic equation". The replicator dynamic equation describes the evolution of the system in terms of the proportion of a selected strategy being played in the population. Moreover, for a selected strategy s_i, if the replicator dynamic equation satisfies $\frac{d}{dt}p_i(t) = 0$, it means that the evolution has been stopped, and the system is in a stable state. We call this a "fixed point" of the replicator dynamic. Its can be proven that the fixed point of the replicator dynamic equation refers to the stable state of the evolution, which is correlated to the ESS.

Some times, a mutant can make a disturbance to the population's strategy, but the disturbance usually can be restored after a few generations. If the above situation happens, we say that this is not a strict ESS, and refer to it as an Evolutionary Stable State. In evolutionary games, the evolutionary stable state may more widely exist than the ESS, and it can also show the stable state of the dynamic of evolution. In this thesis, the term ESS has been used to refer to the evolutionary stable strategy.

With respect to the game's payoff, the replicator dynamic can measure the adoption of the strategies over the entire population. However, although the replicator dynamic equation can represent the dynamics of the evolutionary system and calculate the ESS if it exists, it does not consider the influence of the population structure. Different graph topologies may affect the strategy adoption during the evolution. To understand the effect of the graph topology on evolutionary games, we need to first introduce graph theory concepts.

2.4 Introduction to Graph Theory

Graph Theory was developed to study the structure and properties of graphs. It is commonly recognized that Leonhard Euler first introduced graph theory in his paper about the seven bridge problem [13]. Since then, the study of graph theory has become popular among mathematicians. The concept of "topology" then have been introduced to study the shape of the graph. Many graphs with certain topological structures have been developed later, such as rings and trees, etc.

A graph can be denoted as $G = (V, E)$, where G refers to the graph, V and E refer to the vertices set $V = \{v_1, v_2, ...v_N\}$ and edge set $E = \{e_1, e_2, ...e_K\}$ of the graph respectively. In terms of graph property, there are many ways to categorize graphs, including:

1. Directed and undirected graphs: The edge of a graph may be defined as a directed edge or an undirected edge, depending on whether one can go from one node to another through the edge regardless of the direction or not. If the edges of a graph are all directed, then the graph is a directed graph; if all the edges are not directed, the graph is a undirected graph. There are also mixed graphs where some edges are directed while others are not.

2. Weighted and unweighted graph: Some times, passing some edges may cost more (or less) effort than others, we call graphs with those "unequal weighted" edges are weighted graphs. The cost on each edge is the "Weight" of that edge.

3. Connected and disconnected graph: Regardless of the size of the graph, if at least one path exists between all pairs of nodes in a graph, that graph is a connected graph, otherwise, the graph is a disconnected graph. A connected graph will never include an isolated community.

4. Regular Graph and Complete graph: If all vertices in a graph have the same number of connections (edges to other nodes), then the graph is a regular graph. If a graph

has the maximum number of edges it could have, then the graph is a complete graph. A complete graph is also a regular graph.

Before we start to introduce graph structures, we need to introduce some different approaches to representing graphs. Besides drawing the graph graphically, in mathematics and computer science, there are many different approaches to represent graphs. One of the most important approaches is the graph adjacency matrix. The adjacency matrix of graphs can be used to calculate many important features of the graph.

Adjacency Matrix of a Graph

A graph's adjacency matrix is a matrix that represents the connections between vertices in that graph. Specifically, for a finite graph $G = (V, E)$, the adjacency matrix A is a $V \times V$ matrix. The entry of the adjacency matrix e_{ij} represents whether there are edges between vertex v_i and v_j.

Example of adjacency matrix is demonstrated in Figure 2.2

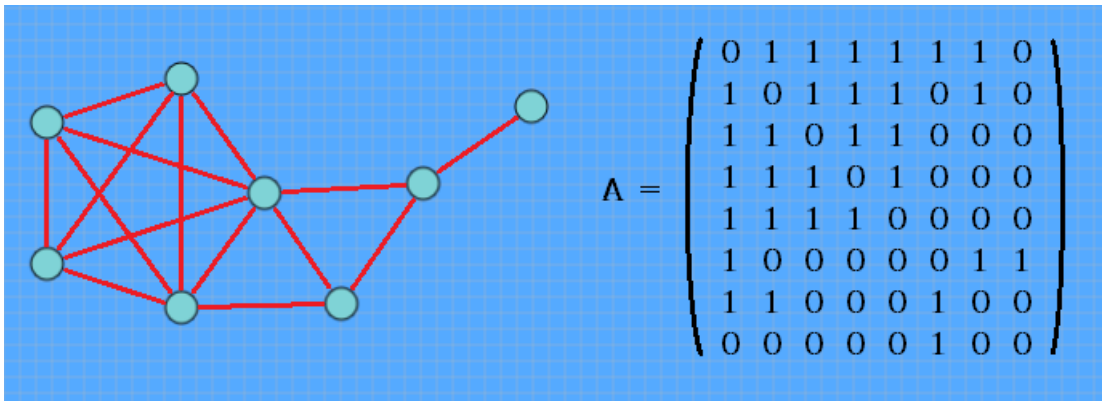

Fig. 2.2 Adjacency Matrix

2.4.1 Topology of The Graphs

The topology of a graph is normally linked to the physical or logical structure of the vertices and edges in the graph. Graph topology has been widely used especially in networks analysis. It also been called "Network Topology" in some of the literature.

There are many complex graph topologies [Figure 2.3] that have been introduced besides the basic network topology. These include, among others:

- Spatial Graph (Lattice Grid) [79–82] : The graph which is connected based on the physical distance between the nodes.
- Newman-Watts Small World Network [126, 127] : Each node in which graph may not connected directly, but it only takes a few steps to reach any node from any other node.
- Scale Free Network [7] : The degree distribution in the graph follows a power-law distribution.
- Erdos-Renyi Random Graph [30] : A graph such that every pair of nodes may connected with a certain probability p.

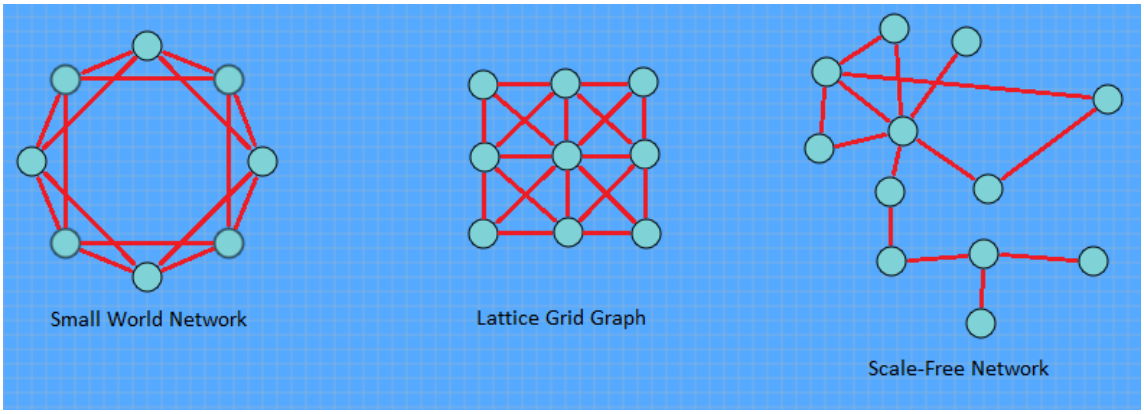

Fig. 2.3 Example of Complex Graphs

It is worth noting that in a graph, especially the scale-free network, some nodes may have a much higher degree (number of edges) than other nodes; those nodes are normally called “Hubs” [7]. To identify the difference in the nodes in the graph, and measure the importance of a node in the graph, the concept of “graph centralities” have been introduced.

In graphs with certain topologies, clusters may exist among the population, which are usually called “communities”. Similar communities may have similar features in terms of the topology. By identifying the same community structure in large graphs, we usually can break the large graphs into small patterns, which is quite helpful for our further research.

2.4.2 Communities

If the nodes in a graph can be divided in to groups according to their connection to other nodes, we call such a group a “community” in the graph. The concept of a graph community

can help us to divide graphs into several sub-graphs, which may allow us to understand the entire graph by studying each sub-graph (usually much smaller than the original graph) separately.

In large (or infinite) graphs, community structures are often being studied instead of the entire graph population structure. There are many community detection approaches to identify communities in graph [33]. The common community detection algorithms usually identify communities in several structures including: clique, club, and clan [72]. The definition of clique, club and clan are quite similar.

Clique and N-Clique

If a sub-graph is a complete graph (which means there exist an edge between every pairs of nodes in the sub-graph), then the sub-graph can be referred as to a "clique". However, the clique is usually very rare in random graphs.

Define 1 *For a graph $G = (V,E)$, there is a maximum sub graph g of G that satisfies: for all pairs of points u,v in g, the distance in G is :*

$$d_G(u,v) \leq N$$

we call the sub-graph g an "N-Clique".

The N-Clique's condition is more likely to be satisfied in random graphs as N increases. When $N = 1$, the 1-clique is a clique, so we can say the clique is a N-clique when $N = 1$. It is important to note that the distance we used to calculate the N-clique is the distance in the graph G, not the distance in the sub-graph g, which means that the distance of node u and v, may be greater than N in sub-graph g, or even u and v may not connected in g, however, as long as their distance is smaller than N in graph G, they could in the same N-clique. By adding additional restricts in to above concept, "N-clan" and "N-club" have been introduced [72]:

N-Clan

The N-Clan g is an N-Clique that also satisfies for each pair of nodes in g, the distance in g is smaller than N :

Define 1 *Given an N-clique g in graph G, ie, $d_G(u,v) \leq N$, if it also satisfies:*

$$d_g(u,v) \leq N$$

then g is called an "N-Clan".

If we only consider the restriction of d_g regardless of the restriction of d_G of N-Clique, we have an N-Club:

N-Club

Define 1 *If any sub-graph g satisfies :*

$$d_g(u,v) \leq N$$

then g is called an "N-Club".

It is easy to see the main difference between N-Clan and N-Club is whether the graph satisfies $d_G\{u,v\} \leq N$. Although it may appear that a sub-graph is an N-clan if and only if it is an N-club, Mokken proved that a sub-graph is an N-club is not the sufficient condition for that sub-graph being a N-clan [72].

Community detection may help us to simplify large graphs into smaller patterns, however, it only decreases the size of the graph. In the research of graph theory, the graph centralities and properties are also important features to understanding the structure of graphs.

2.4.3 Graph Centralities and Properties

In graph theory, the importance of each vertex usually needs to be measured. The measure may be based on how many direct connections a node has, or whether a node's neighbours are more likely to be connected to each other etc. The term "Graph Centrality" have been used to describe those measures. Nodes in graphs have different centralities which may influence their behaviour under certain situations, especially when we are trying to adopt the graph into other domains such as social networks, and evolutionary games. Considering the entire population's centralities in a graph, we can define the properties of the graph.

The idea of graph centrality was first introduced as the Bavelas centrality index to analyse the human community by Bavelas in 1948 [11]. The research of graph centralities has become popular [12, 35, 105].

The ordinary graph centralities include degree centrality, clustering coefficient centrality, closeness centrality, betweenness centrality and eigenvector centrality.

Degree Centrality

Degree Centrality, or node degree refers to the number of connections from a node to other nodes in the graph; the average number of all nodes' degrees is usually referred to used as the degree of the graph [126]. The degree centrality shows the state of connections in graphs, and it is one of the most commonly used centrality measures in graph theory. Many network models are build on the degree centrality; for example, if the degree of every nodes in the graph are same, this graph is called "regular graph"; if the degree distribution of the graph is follows the power-law distribution, this graph is called "scale-free network", in which the high degree nodes are called "Hubs"[7]. The degree centrality is very easy to calculate; by using an edge vector to store the edge, we can calculating the degree for every node in a graph only takes $O(E)$ instead of $O(V^2)$ ($E <= V^2$) time complexity, where E is the number of edges in the graph and V is the number of vertices in the graph. An example of the degree centrality is shown in Figure 2.4.

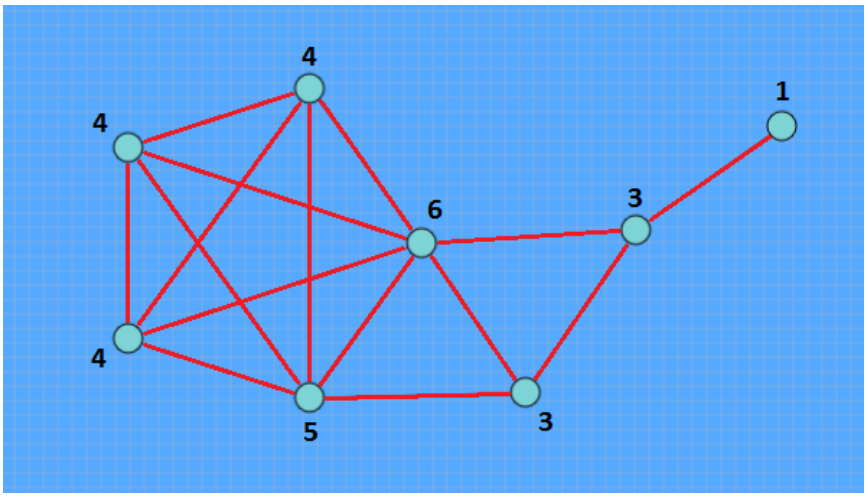

Fig. 2.4 Example of Degree Centrality

Clustering Coefficient Centrality

The clustering coefficient measures the proportion of the number of a node's neighbours that are connected to each other [126]. It is scale from 0 to 1 where 0 means none of the node's neighbours are connected to each other or the node has less than 2 neighbours; and 1 means that all of the node's neighbours are connected to each other. Unlike the node degree centrality, the clustering coefficient only measures the connectivity in the node's neighbourhood (all nodes that connected to the specific node), in other words, the clusters

in graph. So, a low degree node may still have a high clustering coefficient if all of its neighbours are connected to each other, and vice versa.

The term "clustering coefficient" is usually used as the average clustering coefficient of all nodes in the graph [126], for individual node's clustering coefficient, the term "local clustering coefficient" is often used instead. The definition of the local clustering coefficient for a node v_i is :

$$LCC(v_i) = \frac{The\_number\_of\_edges\_in\_v_i's\_neighbourhood}{The\_maximum\_potential\_number\_of\_edges\_in\_v_i's\_neighbourhood}$$

To calculate the clustering coefficient, consider a graph $G = (V,E)$, where $V = \{v_1, v_2, ...v_N\}$ and $E = \{e_1, e_2, ...e_K\}$, the neighbourhood N_i for vertex v_i can be defined as :

$$N_i = \{v_j, e_{ij} \mid v_j \in V \wedge e_{ij} \in E\}$$

then, the number of vertices in neighbourhood N_i is:

$$n_i = \mid \{v_j \mid v_j \in N_i\} \mid$$

and the number of edges in N_i is:

$$k_i = \mid \{e_{jk} \mid e_{jk} \in N_i\} \mid$$

The definition of local clustering coefficient $LCC(v_i)$ for node v_i is:

$$LCC(v_i) = \frac{2 \cdot k_i}{n_i(n_i - 1)}$$

which can also be written as

$$LCC(v_i) = \frac{\lambda_G(v_i)}{\tau_G(v_i)}$$

where $\lambda_G(v_i)$ is the number of triangles that have v_i as any of its 3 vertices, and $\tau_G(v_i)$ is the number of triplets that have node v_i as one of its vertices.

An example of the clustering coefficient can be found in Figure 2.5

The global clustering coefficient can be calculated by taking the average of all individual local clustering coefficients; this was introduced by Watts and Strogatz [127], in which they call it the network average clustering coefficient. Another approach to measure the level of

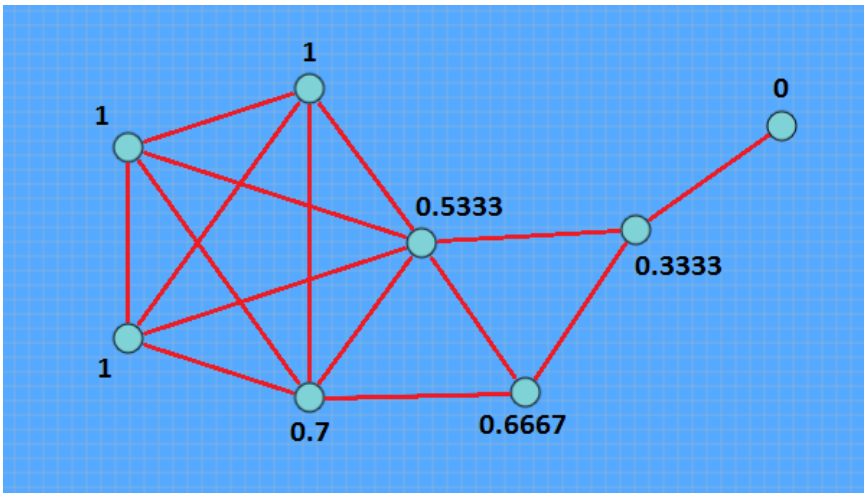

Fig. 2.5 Example of Clustering Coefficient Centrality

clustering is the "Transitivity" measure [84, 100], which can be calculated by:

$$LCC(v_i) = \frac{3 \cdot \lambda_G(V)}{\tau_G(V)}$$

where $\lambda_G(V)$ is the number of triangles in graph G, and $\tau_G(V)$ is the number of triplets in graph G. The transitivity is in positive proportional to the average clustering coefficient, only has slight differences in the value. In this research, we use transitivity as the measure of the clustering coefficient.

Closeness Centrality

"Closeness" is a measure that takes the inverse of the "farness" (which is the sum of the shortest distance from a node to all other nodes in the graph) [12, 105]. For graph $G = (V, E)$, where $V = \{v_1, v_2, ...v_N\}$ and $E = \{e_1, e_2, ...e_K\}$, we can define the shortest distance from node v_i to v_j as $d(i, j)$, and the closeness centrality for node v_i ($CL(v_i)$) can be written as:

$$CL(v_i) = \frac{1}{\sum_{\substack{0<j<N \\ j \neq i}} d(i, j)}$$

Due to the fact that the degrees of every node are different, closeness measured in the above approach is usually a very small value, which makes it difficult to compare across

nodes. A normalised closeness is usually been used instead, which is defined as follow:

$$\| CL(v_i) \| = \frac{N}{\sum_{\substack{0<j<N \\ j \neq i}} d(i,j)}$$

An alternative measure of closeness have also been introduced by Stephenson and Zelen in 1989, which is called the "information centrality" [115]. It can be written as:

$$ICL(v_i) = \sum_{\substack{0<j<N \\ j \neq i}} 2^{d(i,j)}$$

The complexity of calculating closeness is mainly dependent on calculating the shortest distance between every two nodes in the graph, which has a time complexity $O(V^3)$ (Dijkstra's algorithm has time complexity $O(V^2)$, which is single sourced, adopt it to V vertices, which is $O(V^3)$).

Example of the closeness centrality values in a graph can be found in Figure 2.6

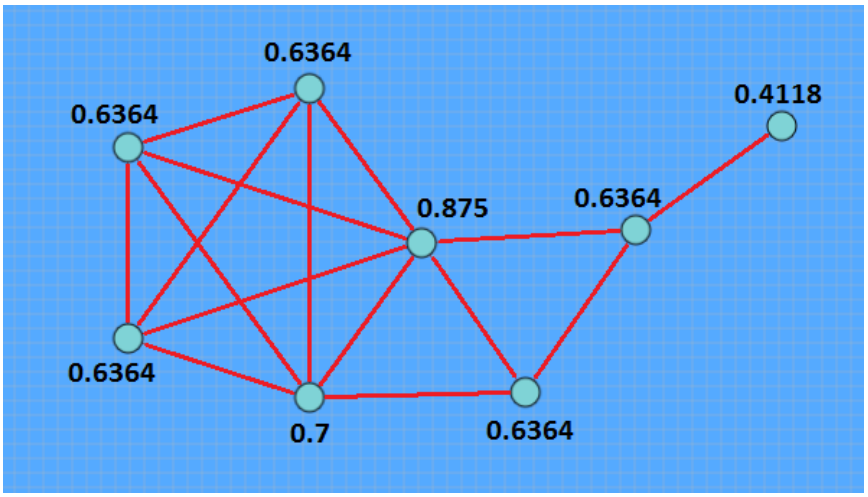

Fig. 2.6 Example of Closeness Centrality

The closeness of a node describes the "position" of the node in graph, whether it is near the "centre" of the graph or not. If a node is near the "centre" of the graph (which means the distance from this node to any other node in the graph are short enough), the closeness score is small, if it far from the "centre" of the graph, the closeness score is big.

Betweenness Centrality

The betweenness centrality of node v_i is the number of shortest paths between all pairs of nodes in graph G that have to pass node v_i. It is first introduced by Freeman to demonstrate the importance of a node, in terms of whether it occupied a position on the shortest path between others [34].

In the graph $G = (V,E)$, where $V = \{v_1, v_2, ...v_N\}$ and $E = \{e_1, e_2, ...e_K\}$, define σ_{st} as the number of short paths between node v_s and v_t, $\sigma_{st}(v_i)$ is the number of short paths between node v_s and v_t, which have passed through node v_i. The betweenness of node v_i is defined as follow:

$$B(v_i) = \sum_{s \neq t \neq v} \frac{\sigma_{st}(v_i)}{\sigma_{st}}$$

Similarly to the closeness centrality, the complexity of calculating betweenness is mainly due to calculating the shortest path between each pair of nodes in the graph. However, by considering that "a vertex is definitely on the path to its successors in the tree of shortest paths from the source" [21], Brande first introduced the "dependency" $\delta_{s\bullet}(v)$ of a vertex $s \in V$ on another vertex $v \in V$, which refers to the number of the shortest path from vertex s to any other vertex that passed vertex v :

$$\delta_{s\bullet}(v) = \sum_{t \in V} \delta_{st}(v)$$

and he found that the dependency of $s \in V$ on any $v \in V$ obeys :

$$\delta_{s\bullet}(v) = \sum_{w: v \in P_s(w)} \frac{\sigma_{sv}}{\sigma_{sw}} \cdot (1 + \delta_{s\bullet}(w))$$

where σ_{sv} is the number of short paths between node $s \in V$ and $v \in V$.

By adopting the dependency in the algorithm, Brande's approach can calculate the betweenness of unweighted graph within a minimum time complexity $O(VE)$ (which is the currently fastest algorithm to calculate the shortest path in unweighted graphs) [21].

Example of the betweenness centrality can be find in Figure 2.7

The betweenness measures introduced above is also called the vertex betweenness; there are also edge betweenness, which quantifies the number of times for an edge being passed through by the shortest paths between any two nodes in the graph. However, when we refer to the betweenness, we are referring to the vertex betweenness in this thesis.

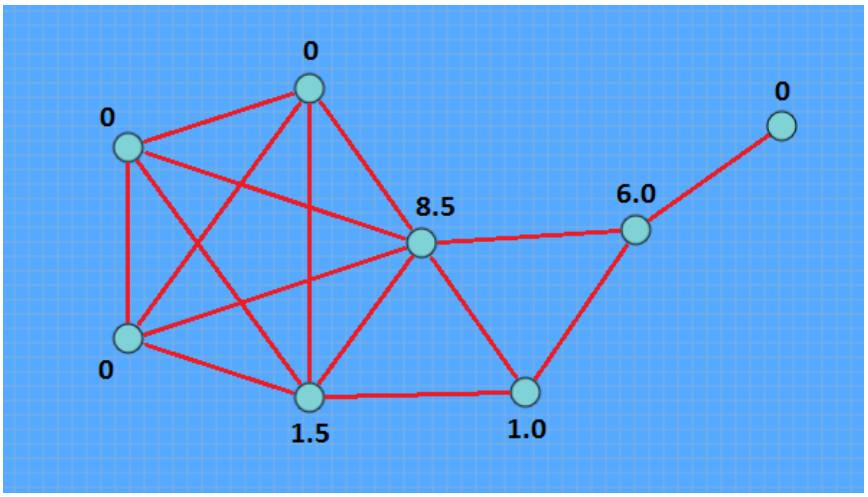

Fig. 2.7 Example of Betweenness Centrality

Eigenvector Centrality

The degree centrality measures the number of edges that a node has; in other words, the degree centrality measures the number of direct neighbours of a node. In some social networks such as linkedin, the second and third connection have also been considered. In this situation, in addition to the direct neighbours of the node, we may also want to know the degree of each of the node's neighbours, and so on. Therefore, the concept of eigenvector centrality has been introduced [14–17].

The eigenvector centrality measures the importance of a node based on its connections to high-scoring nodes in the entire graph. To calculate the eigenvector centrality, the adjacency matrix of the graph $A = a_{V,V}$ is needed. The eigenvector centrality of node v_i in graph $G = (V,E)$ is defined as the corresponding element in the eigenvector e, where $Ae = \lambda e$, and λ is the largest eigenvalue of matrix A.

For the adjacency matrix of a graph, according the Perron-Frobenius Theorem ([114]) :

Perron-Frobenius Theorem 1 *If the largest eigenvalue for an non-negative matrix A, is* λ_{max}*, there are :*

1. λ_{max} *is positive.*
2. *The eigenvector* e_{max} *of eigenvalue* λ_{max} *is unique, and all entries of* e_{max} *are positive.*
3. *If A is primitive (the lagest eigenvalue), then all other eigenvalues of A satisfy:*

$$\lambda_{max} > |\lambda_i|$$

We can guarantee that the eigenvector centrality exists for all graphs.

One of the most common approaches to calculating the eigenvector centrality of graphs is to use the power iteration. Consider a matrix A that has n eigenvectors $e_0, e_1, ..., e_n$. Since the eigenvectors of a matrix are linearly independent, one can set up a initial approximation $e = c_0e_0 + c_1e_1 + ... + c_ne_n$, with a random value of $c_0, c_1, ..., c_n$. Then multiply A by both sides, we have :

$$Ae = c_0(Ae_0) + c_1(Ae_1) + ... + c_n(Ae_n)$$

As e_i is an eigenvector of matrix A, we have $Ae_i = \lambda_i e_i$, where λ_i is the corresponding eigenvalue. Then, we get:

$$Ae = c_0(\lambda_0 e_0) + c_1(\lambda_1 e_1) + ... + c_n(\lambda_n e_n)$$

by repeatedly multiplying A to both side of the formulae, we can get :

$$A^k e = c_0(\lambda_0^k e_0) + c_1(\lambda_1^k e_1) + ... + c_n(\lambda_n^k e_n)$$

presuming λ_d is the dominant eigenvalue of matrix A, in other words, $\lambda_d > \lambda_i, 0 < d \neq i < n$, we get :

$$A^k e = \lambda_d[c_0(\frac{\lambda_0}{\lambda_d})^k e_0 + c_1(\frac{\lambda_1}{\lambda_d})^k e_1 + ... + c_d e_d + ... + c_n(\frac{\lambda_n}{\lambda_d})^k e_n]$$

when k is big enough, $\frac{\lambda_i}{\lambda_d}$ will converge to 0, where we can approximate the value of the eigenvector of the dominant eigenvalue of the matrix A :

$$A^k e \approx \lambda_d^k c_d e_d$$

We can see that when the dominant eigenvalue λ_d is significant lager than other eigenvalue, the convergence will be very fast and we can usually approximate the eigenvector values in under 10 iterations.

This is the central idea of the power iteration; in reality, for the calculation, we can merely initialize a random vector at the beginning of the approximation, and then multiply the matrix to it again and again, and get the next generation of that vector. The vector will converge to the eigenvector of the dominant eigenvalue. This approach has been widely adopted in many algorithms, including the page rank algorithm [91].

Example of the eigenvector have been shown in figure 2.8

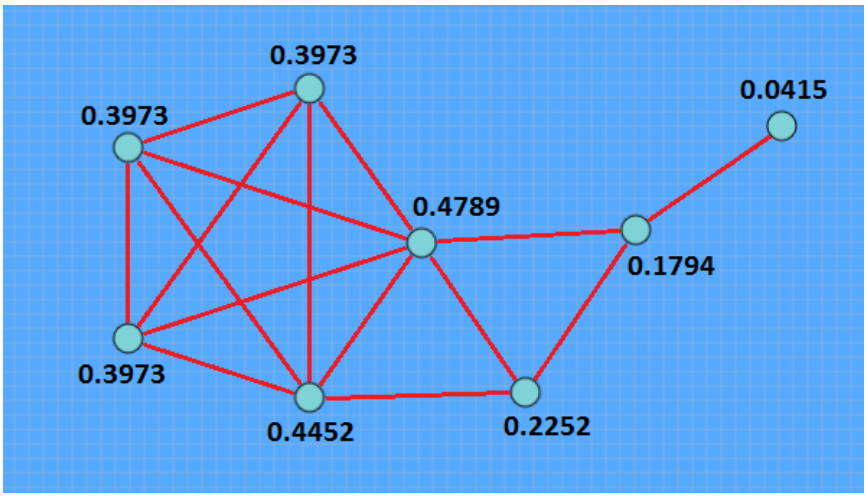

Fig. 2.8 Example of Eigenvector Centrality

Other Centralities

Beside the common centralities described above, there are also many other centralities that have been introduced in different circumstances, such as the game centrality [112], which is used to measure the ability of a defector to spread and invade an almost fully cooperative graph.

So far, we have introduced the notion of different graph centralities and properties. However, to generate a graph with certain properties and centralities, we also need to introduce some graph generation models to generate graphs that we can use in this research.

2.4.4 Graph Generation Models

In order to study graphs with certain properties, we need to have graphs with those properties. Some graphs can be obtained from the real-world. However, it usually takes extensive efforts to gain this real-world data. In many situations, the graphs are generated using graph generation algorithms. Traditional graph generation approaches can generate certain type of graphs, such as Erdos-Renyi model ($G_{n,p}$) [30] which generates graphs by using a constant probability p to create an edge between any two vertices; and small-world model [126, 127], in which the vertices that do not connected directly can be reached from one to another only through a few steps; etc. Moreover, during this research, graphs with tunable properties may also be needed. In this section, we'll introduce some of the graph generation models.

Erdos-Renyi Model and the Random Graph

The Erdos-Renyi Model ($G_{n,p}$) [30] is one of the common models to generate random graphs. It is a probabilistic model where the probability of adding an edge between any two vertices in this model is set to be a constant value p. The probability of a vertex having degree k in the Erdos-Renyi random graph, P_k, follows the binomial distribution, since the probability of adding each edge is independent of the other edges' existence. The Erdos-Renyi Model can be written as below:

$$P_k = \binom{n}{k} p^k (1-p)^{n-k}$$

As the probability p of adding an edge is low, the degree distribution of the Erdos-Renyi random graph can also be approximated by the Poisson distribution.

Barabasi Albert Model and the Scale-Free Network

The Scale-Free network is the graph which follows a power-law degree distribution. The concept of scale-free network was first introduced by Albert-Laszlo Barabasi in 1999 [7]. The scale-free network is identical to the random graph as it includes nodes that have much higher degree than others. The existence of "hubs" is usually considered to strongly increased the robustness of the scale-free network.

The Barabasi Albert model adopts both the concept of growth and the preferential attachment, which means that the network's size will grow over time, and the higher the degree of a node, the more likely the node is to be connected to others. The algorithm usually starts with a initial graph G_0, and keep adding new vertices into the graph. The new nodes will be more likely to connect to the existing nodes with higher degrees. Assuming in the current generation, there are n nodes in the graph, and the degree of node i is k_i, formally, the probability p_i of an existing node v_i being connected by the newly added node is :

$$p_i = \frac{k_i}{\sum_{j=1}^{n} k_j}$$

Watts and Strogatz model and the Small World Network

Another property of note is the property of small world network introduced by Watts and Strogatz [126, 127], which describes a type of graph with nodes that can be reached from other nodes in the same graph within a small amount of steps. The small world network also demonstrates a power-law degree distribution, which is considered to be the major feature of

the scale-free network. A scale-free network doesn't have to be a small-world network since nodes in scale-free networks do not only connect with neighbours within certain steps. The small world property can be found in many real word networks, which include the social networks (such as the famous "six-degree separation") and many biological networks (such as "the brain network" [10]). The study of the small world network becomes more and more popular since the last decade.

The Watts and Strogatz small world model is based on a 1 dimensional lattice grid network (a regular ring), and then add (or replace) an edge of a node with a probability β, to an unconnected nodes. In other words, increasing the randomness of a regular graph by limited amount, we could get a small world network. The probability β can be looked as the randomness of the graph, for the small world network, we have $0 < \beta < 1$, while $\beta = 0$, the graph will becomes a regular graph, and the graph will becomes random graph when β is close to 1.

The Configuration Model

Another iterative approach to generating graphs is that by Molloy and Reed [74]. The approach described is termed a configuration model. Given a certain degree sequence $\{d_1, d_2, ..., d_n\}$ for every node in a graph (to guarantee the existence of a graph, the sum of the degrees must be an even number, which satisfies $\sum_1^n d[v_i]\%2 = 0$). A basic configuration model of graph $G = (V, E)$ can be described as follows:

1. For each $v_i \in V$, make $d[v_i]$ number of copies of the node v_i, which is denoted as ${v_i}^k$, $0 <= k < d[v_i]$.

2. Randomly select two nodes ${v_i}^k$ and ${v_j}^l$ and build an edge between them. Each copy of node can be used only once, and repeat until all copies have been connected to another.

3. Merge or replace the edges according to requirements.

There are many variations of the configuration model, such as the algorithm defined by Newman [77], which allows us to approximate a certain clustering coefficient with a given degree sequence.

Graphs generated by the configuration model normally includes multiple edges and self loops (nodes that are connected itself), the copy of edges and self links need to be removed if we want to generate simple graphs using the configuration model.

The Preferential Attachment Model

The original idea of a preferential attachment model is learned from the citation network [97], which is described as "success generates future successes". The idea have been adopted into the concept of the preferential attachment model by Barabasi and Albert [7].

Algorithms based on the preferential attachment model usually begin with a certain connected set, and then add new nodes to connect to the existing nodes with a probability P based on the node's degree $d[v_\bullet]$, satisfying:

$$P(d[v_i]) = \frac{d[v_i]}{\sum_{j=1}^{n} d[v_j]}$$

Due to the feature that "high degree nodes will get more connections" [7], the preferential attachment model is generally used to generate graphs with power-law degree distribution. It is widely used to generate scale-free networks, which require the nodes' degree distribution to follow a power-law distribution.

2.5 Evolution on Graphs

2.5.1 Graph Topology Effects the Evolution of Cooperation

Spatial Games

Two player iterated games and evolutionary games on complete graphs have been well studied [68, 120, 128]. The ESS of an evolutionary game on a complete graph can be calculated using replicator dynamic functions. Many researchers have drawn attention to particular graph topologies, includes regular graphs [37, 45, 56, 79–82], small-world networks [24, 36, 85, 106, 124], scale-free networks [4, 7, 22, 94, 95, 98, 104, 116, 117, 122, 129], and others.

Nowak and May first simulated the Prisoner's Dilemma games on a two-dimensional spatial graph (or lattice grid) [81, 82], and visualized the evolution graphically (Figure 2.9 displayed the evolution of patterns of a simulation by Nowak and May). Nowak and May set each player in the game to adopt a pure strategy (to cooperate or not). In each generation, the player plays the prisoner's dilemma game with their 8 immediate neighbours and with themselves (Moore Neighbourhood) and then adopts, at the beginning of the next generation, the strategy of the neighbour receiving the highest payoff. Nowak and May plotted cooperators and defectors, strategies who cooperated in the current generation but defected

in the previous generation, and those strategies who defected in the current generation but cooperated in the previous generation in order to visualise the emerging patterns. When the initial population was set to be one defector surrounded by cooperators, the evolutionary game resulted in a dynamic pattern. Nowak and May found that the payoff of a defector against a cooperator, represented as b, is actually spread discretely [79]. The cooperation rate during the evolution will only be changed at some critical value of b. However, these critical points depend on the numbers of direct neighbours of each individual, which make the evolutionary outcomes predictable on a regular graph, such as the lattice grid.

Having analysed the clusters of cooperative and non-cooperative players, Nowak and May performed experiments starting with a randomly initialised population, and tried to observe how the players maintained cooperation over different settings of the value of b[80].

They posit that the different strategies (C and D) could more easily co-exist in spatial games. Researchers believe that the spatial structure could increase the cooperation rate in evolutionary games, as co-operators could gather in a cluster, which not only could prevent invasion by defectors but could also invade the defectors in the remaining part of the graph.

Langer studied the invasion of clusters of cooperators in the PD game [56], and he found that a cluster of cooperators is able to invade a group of defectors. Increasing the defector's payoff could decrease the invasion speed, but the cooperators and the defectors may always find certain stable states to coexist; they refer to one this scenario as "the local equilibrium".

Hauert also studied the spatial effects on social dilemmas [45]. He demonstrated that the spatial structure could benefit the cooperators in the prisoner's dilemma game, but not the Snow Drift game. Then, Fu played both the Snow Drift game and the Prisoner's Dilemma game on spatial graph, and analysed the difference in a cluster of cooperator's ability to invade non-cooperative strategies in the two different games [37]. Fu found and explained that under less favourable conditions, in spatial graphs, the prisoner's dilemma's cooperation rate could be improved, while cooperation was inhibited or could even be eliminated in the snowdrift game.

More Graphs

One of the "Five Rules" Nowak concluded is that the "Network Reciprocity" can influence the evolution of cooperation [78]. Actually, since Nowak and May successfully showed the emergence of cooperation on the lattice grid graph [79], much research has been done to study the influence of the population structure to the evolution. Santos showed that the graph topology can influence the cooperation of evolution [106], in which, the experiments showed the graph generated by the preferential attachment model with heterogeneous popu-

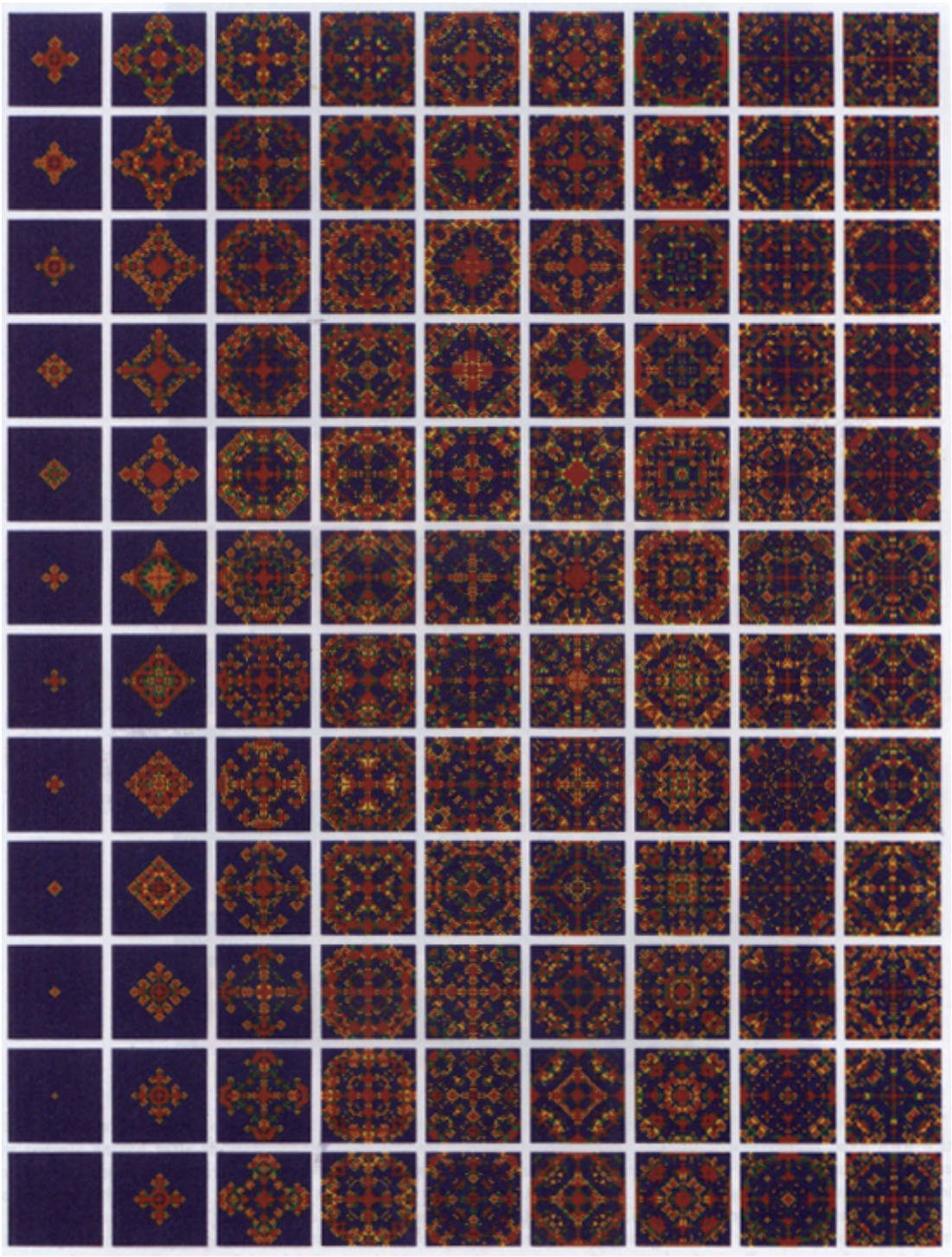

Fig. 2.9 Nowak and May's visualisation

lation (where in the graph, some nodes have a much higher degree than others), may benefit cooperation, due to the hubs that are occupied by the cooperator are more stable than the hubs that are occupied by the defector.

Research has also shown that certain graph topologies such as the scale free and small world networks, may induce more cooperation than others [36, 54, 92, 107, 132]. For example, Santos and Pacheco found that the cooperators could become more competitive or even predominant in scale-free networks [107]. Fu found that, in small-world networks, the degree heterogeneity is critical to the emergence of cooperation; cooperation reaches its peak at some intermediate value of the fraction of hubs and not at either the most heterogeneous nor the most homogeneous case [36]. Moreover, many works have explored evolution dynamics over different configurations of different population structures [25, 37, 38, 45, 46, 90, 122].

It is commonly believed that the good performance of cooperation on the scale-free and small-world network is due to the power-law degree distribution. A graph has the power-law degree distribution is usually called an "heterogeneous graph". Experiments shows that heterogeneous graphs are normally more robust on maintaining cooperation in comparison to homogeneous graphs in the more general situation [95, 108, 109]. There is also evidence showing the power-law degree distribution is not the only property that can benefit the cooperation–for example, cooperation is benefited when one averages the payoff garnered through game interactions [65] and also under weak selection [58]. Poncela found that besides the power law distribution of the node degree, the node to node correlation also plays an important role in the performance of cooperation in a scale-free network [96]. Fukami defines a class of graph, which in their assumption, has the maximum clustering coefficient over all graphs with the same number of vertices and edges [39]. Also, small community-like clusters are also suitable for cooperation to survive, as the cooperation is more robust in graphs that have higher transitivity (or clustering coefficient) but lower average degree.

As heterogeneous graphs have shown their ability to maintain cooperation in comparison to homogeneous graphs in the more general situation [71, 95, 108, 109], research in the domain has turned to exploring the graph attributes (both local and global) that can influence the emergence of cooperation. Community structure and even individual behaviour have also been well studied [44, 51, 73]. One emerging opinion is that cooperators on hubs and in highly connected communities are more likely to survive [64, 103]. Attempting to understand how an individual player can influence the cooperation on the entire graph, Shigaki examined the initialization setting of the population of evolutionary prisoner's dilemma

[110], included the size of cluster, number of cluster, and the shape of the cluster. Experiments show that having more cooperators at the beginning does not necessarily lead to a higher cooperation rate in later generations, because defectors at initial generations in such settings will get higher payoff as they are connected to more cooperators. All of the properties mentioned above have already been considered as a potential features that can contribute to the emergence and robustness of cooperation. Besides, there are even new centrality measures that have been defined such as Banzhaf, Shapley-Shubik, effort and satisfaction centrality [73]. In particular, the spread of defection from an individual player has also been considered as a measure of graph centrality (called "Game Centrality") [41, 42, 112], and this measure has shown its usefulness in several real world networks [112].

2.5.2 The Evolution of the Graph Structure

During the study of the influence of the graph topology on the evolution, researchers have started to let the graph and the game evolve together, and observe the structure of graphs that have been generated during the process of the evolution, such as Majeski, Sicardi and Chiong who studied the IPD (iterated prisoner's dilemma), N-IPD (N-player IPD), SD (snowdrift) and SH (stag hunt) game in a spatial graph with moveable agents, and declared that introducing agent movement into the graph can significantly increase the stability of the cooperation [26, 67, 111].

The concept of the evolutionary graph theory have been proposed by Lieberman, Hauert and Nowak [63] in 2005. The evolutionary graph theory is to let the graph change its structure during the evolution, it is an alternative approach to study the cooperation during the evolution.

2.6 Summary

In this chapter, we have discussed the related work in the domain of graph theory and evolutionary game theory. The important concepts used in this research have been introduced in this chapter. The introduction of the research background included evolutionary computation, game theory, graph theory, and the graph centralities which will be used in the rest of the thesis. We have also introduced some common graph generation model to compare with the heuristic generation model we'll introduce later in the thesis.

Furthermore, by reviewing previous researches, we introduced the currently popular research topic in the domain of evolutionary game theory, and identified the importance and

the motivation of studying the influence of graph topology in the evolution. The following work will took the research above as a fundamental, especially Nowak's "five rules" of the cooperation [78], and then looking for the theory behind each "rules" and trying to identify which specific attribute or centrality can influence the cooperation. The research in this thesis will include the study of the graph structure, community, and individual centrality's effect on the evolution of cooperation.

Chapter 3

Evolutionary Games on a Lattice Grid

3.1 Introduction

One of the most commonly researched topics in evolutionary games is that of evolutionary games on lattice grids. Following Nowak and May's work [79–82], many researchers have repeated the experiments in different settings, and tried to discover how the payoffs of the game may affect the cooperation. The lattice grid is one of the most commonly researched regular graphs, the regularity of the lattice grid can minimize the influence of the graph to the cooperation, and helps us to focus on the influence of the payoff matrix itself. In this chapter, we researched two-player pure strategy games on lattice grids with Moore neighbourhoods, analysed the influence of the payoff matrix to the cooperation rate, and gave predictions regarding the cooperation rate of the evolutionary games with fixed initial settings.

3.2 Payoff Matrix and Payoff Plot

3.2.1 Linear pure strategy games

A two strategy game is a game where each player can pick one of the two available strategies. The payoff of each strategy of each player can be shown in Table 3.1.

Table 3.1 shows two-players, P_1 and P_2, playing a pure strategy game which has two strategies S_1 and S_2. Player P_1 plays as the column player and P_2 plays as the row player. P_1a and P_2a refer to the payoffs of P_1 and P_2 respectively when P_1 plays S_1, and P_2 plays S_1, and so on.

$P_1 \backslash P_2$	S_1	S_2
S_1	$P_1 a \backslash P_2 a$	$P_1 b \backslash P_2 c$
S_2	$P_1 c \backslash P_2 b$	$P_1 d \backslash P_2 d$

Table 3.1 Payoff of a two-player game

For a single player, the general two strategy game's payoff matrix can be simplified as follows.

$$P = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Considering the linear pure two-strategy games first, the payoff for a player picking a strategy can be represented as a linear function of the choices of the player's neighbours.

The payoffs are plotted in Figure 3.1.

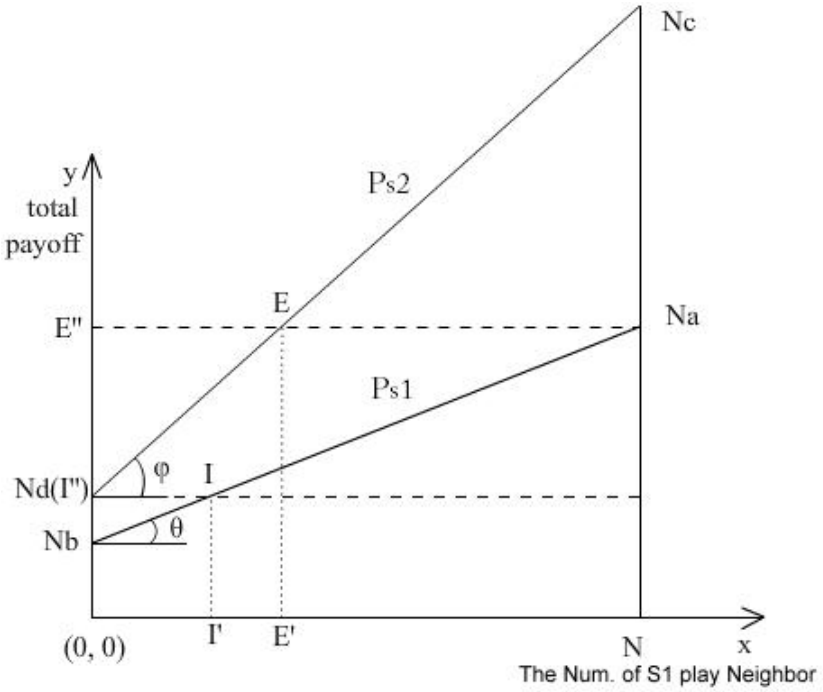

Fig. 3.1 General payoff plot of a two strategy game

In Figure 3.1, the number of neighbouring players playing S_1 is represented on the x-axis, and the total payoff obtained is represented on the y-axis. There are two payoff functions displayed in the graph. One line represents the payoff obtained by a player playing

strategy S_1 and is calculated according to: $P_{s1} = N \times b + (a-b) \times x$. The other line represents the payoff obtained by a player playing strategy S_2 which is calculated according to: $P_{s2} = N \times d + (c-d) \times x$. For both formulae, the value is calculated for a neighbourhood of $N+1$ players (include the player itself).

It is worth noting that, for a social dilemma game, the following relations must be satisfied:

$$\begin{cases} \theta = \arctan(a-b), 0 < \theta < \pi/2 \\ \phi = \arctan(c-d), 0 < \phi < \pi/2 \\ d \leq a \leq c \\ 2a \geq b+c \end{cases}$$

We can define a function $P_N(x)$ (referring to the maximum value obtained by a strategy given the choices of a strategy's neighbours) as follows:

$$P_N(x) = \begin{cases} P_{S1}(x), P_{S1}(x) > P_{S2}(x), \\ P_{S2}(x), P_{S2}(x) > P_{S1}(x) \end{cases} \quad 0 \leq x \leq N$$

This function represents the Nash equilibrium for each point. Given x players playing strategy $S1$, $P_N(x)$ represents the choice corresponding to a Nash equilibrium given N neighbours.

The Nash equilibrium in this case is the pure strategy equilibrium. For the prisoner's dilemma, it is easy to see that the strategy of choosing to defect is the Nash equilibrium for the game. However, in some games, such as the snow drift game and stag hunt game, it is not as simple. We can see that there exists a value m which makes the game have two separate pure strategy Nash equilibria. For the stag hunt game, there are $(R > T > P > S)$, leading to the following plot of payoffs (Figure 3.2):

If we define strategy S_1 as cooperate, and strategy S_2 as defect, then:

$$P_N(x) = \begin{cases} P_D(x), 0 < x < m \\ P_C(x), m < x < N \end{cases} \quad , 0 \leq x \leq N$$

where $P_D(x)$and $P_C(x)$ refer to the payoffs received by those strategies choosing D and C respectively.

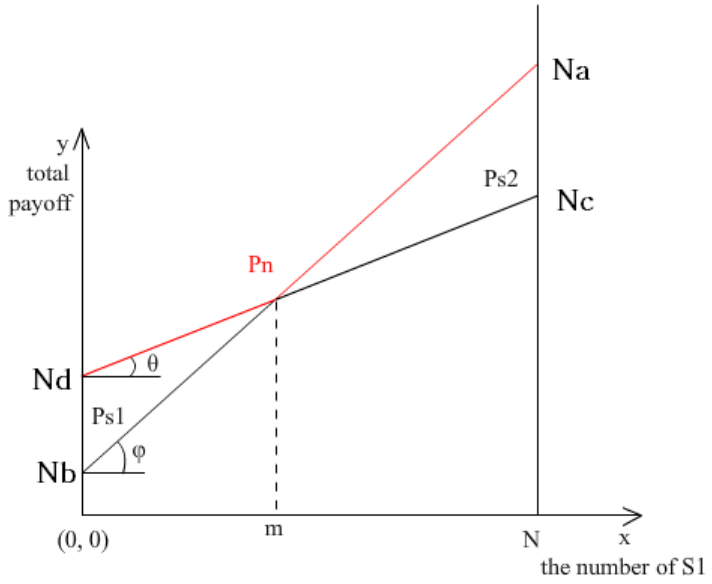

Fig. 3.2 Nash equilibrium payoff plot

Similarly, for the snow drift game, we have:

$$P_N(x) = \begin{cases} P_C(x), 0 < x < m \\ P_D(x), m < x < N \end{cases} \quad ,0 \leq x \leq N$$

For a social dilemma game, if the game has only one pure-strategy Nash equilibrium, then the Nash equilibrium is to defect (D) (such as prisoner's dilemma). If the game has two pure-strategy Nash equilibria, then there exists a mixed-strategy Nash equilibrium if we allow mixed strategies; we will consider this mixed strategy Nash equilibrium later.

An evolutionary stable state (ESS) is a state where the composition of the number of players adopting each strategy is stable; in other words, a single mutator will not affect the proportion of the strategies that are adopted by individuals in the population in the evolution. However, unlike the games in the complete graph, it is hard to calculate the ESS for games on more complex networks. Current research typically adopts agent-based simulation to simulate such evolution, and then searches for the ESS if there is one, or analyses the number of individuals that are playing the different strategies.

If a player which played strategy S_1, adopts strategy S_2 in the next generation, we say that strategy S_2 invades strategy S_1 on this vertex of the graph.

Given N two-player games, the payoff function of both cooperator and defector are

linear functions. We find that the point of intersection of $x = 0$ (number of cooperative neighbours), and $x = N$ is the minimum and maximum payoff of the cooperator and defector respectively. We can find the numbers of cooperative neighbours (the x axis) that strategy $S1$ needs to reach the minimum and maximum payoff of strategy $S2$. It is easy to see that $S1$ can only invade $S2$ in this section. By drawing horizontal lines at the maximum and minimum payoff, we can find those points (for example, I, and E on Fig. 3.1), we refer these two points as the "invade point"(I_{s2}) and the "escape point" (E_{s2}) of $S2$ (which means that only between those two points may $S2$ be invaded by $S1$).

In social dilemma games, in the section $x \in [0,N]$, only defectors have a escape point (because the escape point of the cooperator's x will be greater than N, since b is always greater than 1). Also, we may find that one of the x-axis values for a strategy's ($S1$ for example) invade point may be smaller than 0 (cooperator for prisoner's dilemma, defector for snow drift, etc.). We say that $S1$ can be invaded by $S2$ from $x_{s2} = 0$, and $S2$ can be invaded by $S1$ from $x_{s1} = x_{I_{s2}}$.

For example, in the prisoner's dilemma's payoff plot (Fig. 3.1), there is a point I on the payoff line of strategy S_1 (P_{S1}), and a point E on the payoff line of strategy S_2 (P_{S2}). We label their projection on the X axis as I' and E', and their projection on the Y axis as I" and E".

For a player who is playing strategy S_1, if it has n neighbours playing S_1, it can invade a player that is playing strategy S_2 that has less than m ($m = P_{S2}^{-1}(P_{S1}(n))$) neighbours playing S_1. We know that $0 \leq m$ and $n \leq N$. When $n = I'$, $m = 0$, so the point I is the earliest point for when a player playing an S1 strategy can invade a player playing the S_2 strategy.

We refer to point I as the "invade point" for strategy S_1. When $m = E'$ and $n = N$, this means that point E is the last point that a player playing S_1 strategy can invade a player that is playing the S_2 strategy. In other words, a player who is playing the S_2 strategy that has more than E' neighbours playing S_1 strategy, can never be invaded by any player playing the S_1 strategy. We call this point E the "escape point" for strategy S_2. Conversely, for a player who is playing strategy S_2, if it has more than E' neighbours playing strategy S_1, it could invade any players playing strategy S_1.

3.2.2 Non-linear game and mixed strategy

A game may be linear or non-linear, i.e., the payoffs awarded to players do not need to be a linear function of the number of strategies adopting a particular strategy (for example, a N-player game may have a non-linear payoff function). In many social dilemmas, it makes

sense to model the interaction as a linear function. In many other situations, this is not the case; e.g. people not smoking in a non-smoking room—once one person chooses to smoke, it can be argued that a positive payoff is lost for all.

A non-linear game may have more than one invasion and one escape point. The space of non-linear games may be far more complex than that of linear games. Fig. 3.3 presents an example with several invasion and escape points noted for each strategy.

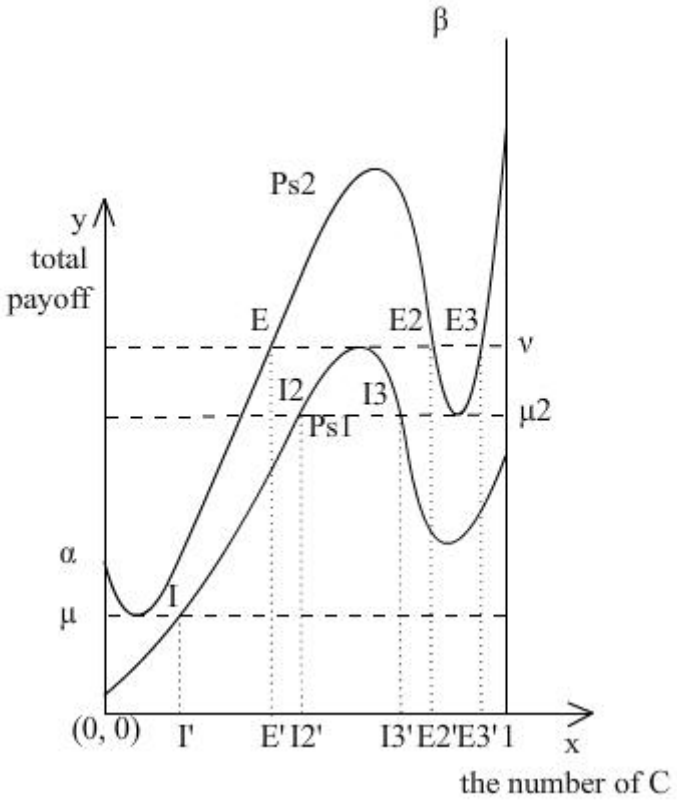

Fig. 3.3 Non-linear game's payoff function

However, we can still calculate every invasion and escape point. We can see, from Fig. 3.3, that the invasion points of strategy S_1 are related to the minima of the payoff function for strategy S_2. Similarly, the escape points for strategy S_2 are related to the maxima of the payoff function for strategy S_1.

Assume that P_{S1}, the payoff function of strategy S_1, has m maxima which are ν_i, $0 < i \leq m$, and that P_{S2}, the payoff function of strategy S_2, has n minima which are μ_j, $0 < j \leq n$. If P_{S1} and P_{S2} are known, we know that P_{S1} and P_{S2} are continuous, the extrema of P_{s1} and P_{S2} are easy to calculate. We can calculate the invasion and escape points using the following

formulae:

$$\begin{cases} I'_j = P_{S1}^{-1}(\mu_j) \\ E'_i = P_{S2}^{-1}(\nu_i) \end{cases}$$

The focus of the research in this chapter is based on the invasion and escape points; if these points of the game can be calculated, we believe the non-linear game could be generalized from the result of the linear game that we introduce next.

A mixed strategy game does not overly differ from a pure strategy game. We can visualise and represent the payoffs for a mixed-strategy as a line which lies between the lines representing the payoffs of the two pure strategies from which the mixed strategy is derived. This is shown in Fig. 3.4 where $S1$ and $S2$ are the two pure strategies, $S3$ is a mixed strategy and the lines denoted $P_{Si}, i \in \{1,2,3\}$ are the payoffs obtained.

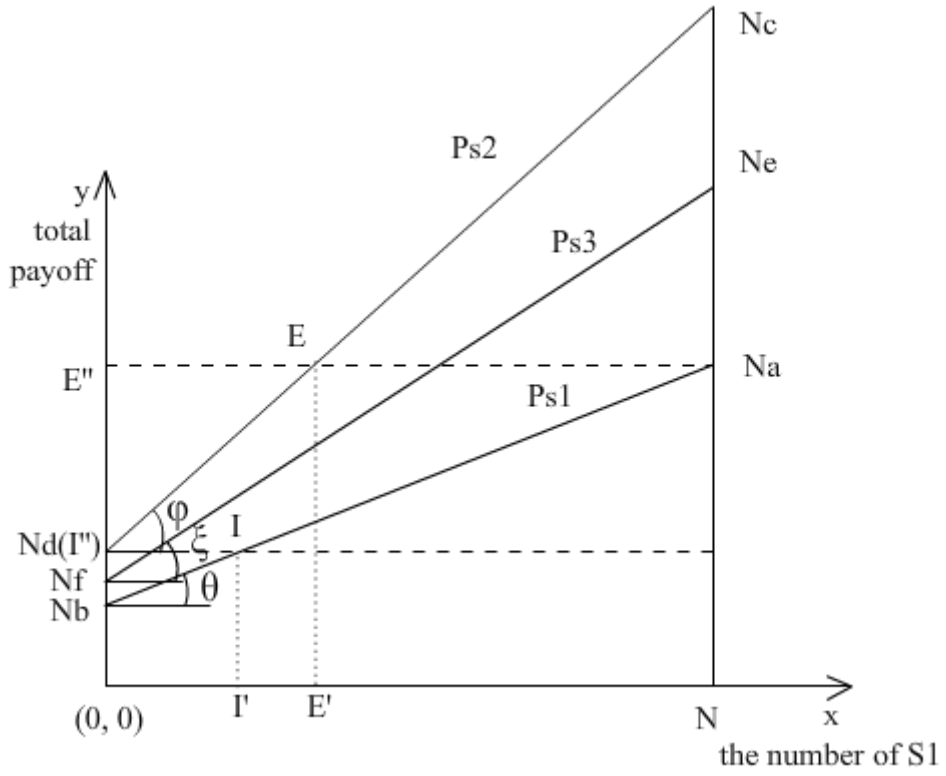

Fig. 3.4 Mix strategy

Notice that the payoff of the mixed strategy S_3 must be between the payoff of the two pure strategies S_1 and S_2. We can also calculate the invasion and escape points of the mixed strategy S3 using the same approach as explained for the pure strategy. For example, in Fig. 3.4, assume that strategy S_3 is $S3 = pS1 + (1-p)S2$, (the probability p follows $0 \leq p \leq 1$).

We have $e = pa + (1-p)c$, and $f = pb + (1-p)d$

Strategy S_3 has the payoff function $P_{s3} = N \times f + (e-f) \times x$. ($x$ is the value of x axis(the number of cooperate neighbours))

Then:

$$\begin{cases} p + (1-p) = 1 \\ \frac{p}{1-p} = \frac{e-a}{c-e} = \frac{f-b}{d-f} \end{cases}$$

and

$$\begin{cases} \theta < \xi < \phi & \text{if } \theta < \phi \\ \phi < \xi < \theta & \text{if } \phi < \theta \end{cases}$$

In some games such as the snow drift game, and the stag hunt game, there are two pure-strategy Nash equilibria. There exists a mixed strategy Nash equilibrium P_{EMS} for those games. For all the mixed strategies of those games, the mixed strategy Nash equilibrium P_{EMS} is the one which obtains the maximum total payoff of all the mixed strategies P_{MS}, which means it should satisfy $\int_0^N P_{EMS} = \max(\int_0^N P_{MS})$.

For now, we only focus on the linear pure strategy games, which is easier for us to do the prediction.

3.2.3 Normalization

As we mentioned before, there are many different games that have been researched such as the prisoner's dilemma, the snow drift game (hawk and dove game), the stag hunt game etc. Some researchers are exploring the similarities and the differences between those games ([25, 37, 66]). However, if we could find a general form for those games, we could probably understand the reasons underlying the differences in the emergence of cooperation in different games.

To simplify the problem, many researchers have tried to scale the games in different ways [56, 108, 119]. In this research, from the perspective of simplifying the payoff function, we adopt a normalization which is similar to Santos' approach.

By applying a basic transformation on the payoff function, the payoff matrix of two strategy game can be normalized.

For any two $\times$ two payoff function like:

$$P = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

The payoff functions are:

$$\begin{cases} P_{S1} = N \times b + (a-b) \times x \\ P_{S2} = N \times d + (c-d) \times x \end{cases}$$

The 2 dimensional coordinates $((x,y))$ of invasion point I for strategy S_1 and the escape point E for strategy S_2 can be represented as:

$$\begin{cases} I = (\frac{N \times (d-b)}{a-b}, N \times d) \\ E = (\frac{N \times (a-d)}{c-d}, N \times a) \end{cases}$$

The calculation of I and E is quite complex in the general situation. To simplify, we need to normalize the payoff function, since there are too many variables. By doing a very basic 2D transformation, we can normalize the formula to be a much simple form, just by setting $N^* = 1$, $\theta^* = \pi/4$, and $\phi^* = \phi \times \frac{\pi}{4\theta}$. So, $a^* = 1$, $b^* = 0$, $\beta = c^* = c \times \frac{\pi}{4\theta} - b$, $\alpha = d^* = d - b$. This transform does not affect the value of payoffs nor the levels of cooperation; it only simplifies the further calculation.

The normalized payoff function becomes:

$$\begin{cases} P_{S1} = x \\ P_{S2} = \alpha + (\beta - \alpha)x \end{cases}$$

The payoff matrix following normalization is similar to Santos' approach [108]:

$$P = \begin{pmatrix} 1 & 0 \\ \beta & \alpha \end{pmatrix}$$

The plot of the payoff function is shown in Figure 3.5.

The normalization doesn't change the property of the original game. It reduces the number of two strategy games, and reduces the number of variables from four to two, which significantly simplifies the calculation of I' and E' (the value of the x coordinate of point I and E):

$$\begin{cases} I' = \alpha \\ E' = \frac{1-\alpha}{\beta-\alpha} \end{cases}$$

where $-1 < \alpha < 1$, $1 < \beta < 2$, and $0 < \phi < \pi/2$ ($\beta - \alpha > 0$).

Most importantly, it gives us a general model which can be used to form different games. For example, in the two-player prisoner's dilemma, the defector's pay off always higher than

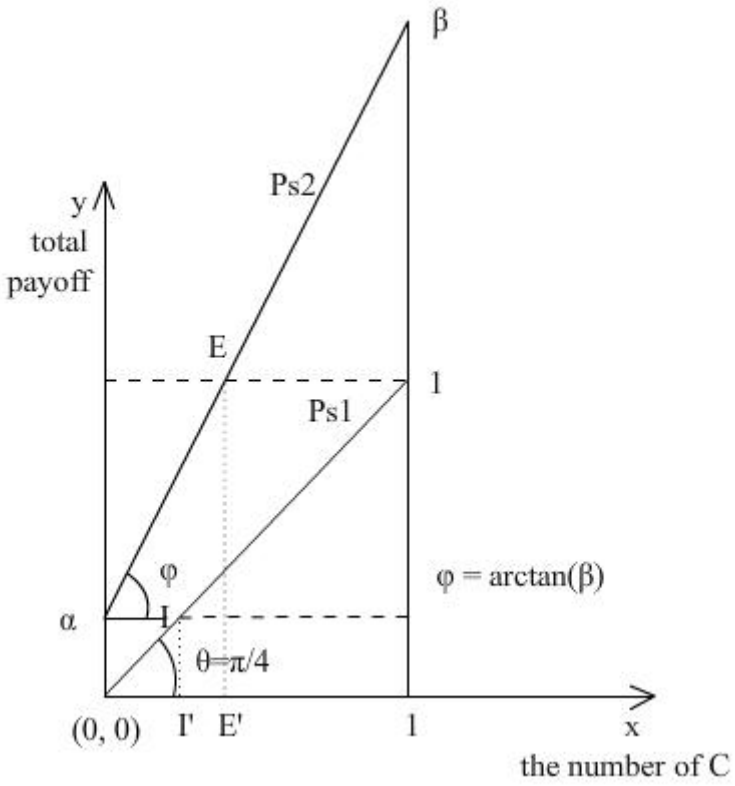

Fig. 3.5 Normalized payoff plot

the cooperator, however, in our experiments, the player will not change it's strategies if they have same payoff with others. In this case, the payoff matrix of prisoner's dilemma can be reducted to:

$$P_{2PD} = \begin{pmatrix} 1 & 0 \\ \beta' & 0 \end{pmatrix}$$

Similarlly, the two-player snow drift game's payoff matrix can be represented as:

$$P_{SD} = \begin{pmatrix} 1 & 0 \\ 1+\gamma & \gamma-1 \end{pmatrix}$$

and the two-player stag-hunt game's payoff matrix can be represented as:

$$P_{SH} = \begin{pmatrix} 1 & 0 \\ \alpha' & \alpha' \end{pmatrix}$$

The parameter α can be greater, equal or less than 0; it must be less than 1. The difference between the above games is only in how α and β affect each other when one of them has been changed. If we can generate a 2D graph to show the emergence of cooperation for different values of the two parameters α and β, we may discover a general attribute underlying the emergence of cooperation in all two strategy games. Furthermore, if the emergence of cooperation is always the same during the evolution with the same parameter settings for a fixed initial generation, we could use this property to roughly predict the result of the evolutionary game given a certain initial generation.

3.2.4 The invasion of strategies

In this research, players are set to adopt the strategy of the neighbouring strategy who obtained the best payoff in the previous generation. Hence, a player's strategy that obtains a high payoff score may replace the other strategies in the neighbourhood. We refer to this as a strategy *invading* another strategy. In social dilemmas, strategy S_1 refers to cooperate, and strategy S_2 refers to defect. There are two possible types of invasion in our setting: strategy S_1 may invade strategy S_2 which results in cooperators replacing defectors, and conversely strategy S_2 may replace strategy S_1 resulting in the defectors replacing the cooperators. To ease explanation, we'll use the terms *cooperate* and *defect* instead of strategy S_1 and strategy S_2 later in this section.

Given a particular neighbourhood NH_1 with a defector interacting with N neighbours, assume there are m players adopting a cooperative strategy in the neighbourhood with which

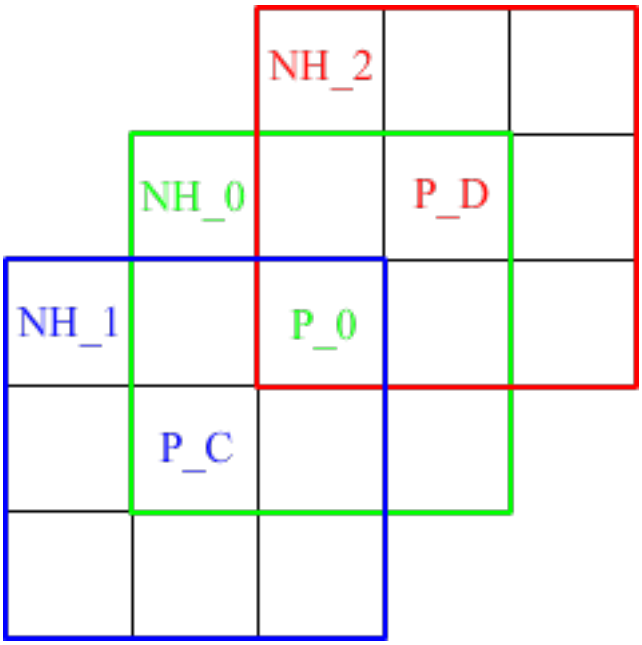

Fig. 3.6 An example of the invasion in the Moore neighbourhood. P_0 is the current player, the cooperator with the highest payoff is P_C and the defector with the highest payoff is P_D. Realise that both P_C and P_D are in P_0's neighbourhood NH_0, but P_C and P_D do not have to be in the same neighbourhood. The minimal number of common neighbour of P_C and P_D is one, which is P_0.

the current player has interacted. The cooperative strategy may be able to invade a non-cooperative (defecting) strategy (which is the Nash equilibrium) if and only if $\frac{m}{N} > I'$, because only when the cooperator has more cooperative neighbours than $m = N \times I'$, may it have a better payoff than a defector with no cooperative neighbours. So, the minimum number of the cooperators in the neighbourhood that are needed is dependant on the value I'.

We know that if $I' = \alpha$, then $\frac{m}{N} - \alpha > 0$. If the game is completely ruled by cooperators, then m has to be 0 (because $m \geq 0$), which means, a cooperator that has no cooperative neighbours, will have a greater payoff than a defector with the same number of cooperative neighbours, then the cooperator can invade a defector to whom he is connected. So this game is a harmony game. The value of α (which is the x value of the invasion point) defines the minimum value that m can take. When N is finite, the invasion points can only be located as defined by the payoff function of the cooperator when:

$$\alpha = \frac{m}{N}, 0 \leq m \leq N$$

On the other hand, for a defector in a neighbourhood (NH_2) which has n cooperative neighbours, if $\frac{n}{N} > E'$, it can invade any of the cooperators as when the defector has more cooperative neighbours than $n = N \times E'$, its payoff will be higher than a cooperator with all of his neighbours are cooperator. We know that $E' = \frac{1-\alpha}{\beta-\alpha}$. Hence, we know that when N is finite, the escape point can only be positioned on the payoff function of the defector when:

$$\frac{n}{N} = \frac{1-\alpha}{\beta-\alpha}$$
$$\Rightarrow N - N\alpha = n\beta - n\alpha$$
$$\Rightarrow N = n\beta + (N-n)\alpha$$

Consider a player, P_0, in its neighbourhood, NH_0, which has c cooperative neighbours ($0 < c < N$). Assume P_C is the player receiving the highest payoff among all of the c neighbours who are playing cooperate. Furthermore, assume that P_C has m neighbours who cooperate in its neighbourhood NH_1. Assume that P_D is the player receiving the best payoff in all of the $N-c$ neighbours who play defect. P_D has n cooperative neighbours in its neighbourhood NH_2.

The player P_1's payoff is:

$$\Pi_C(\frac{m}{N}) = \frac{m}{N}$$

The player P_2's payoff is:

$$\Pi_D(\frac{n}{N}) = \alpha + (\beta - \alpha)\frac{n}{N}$$

We know for a defector to have the same payoff as a cooperator that has m cooperative neighbours, it must have n' cooperative neighbours.

$$n' = N * \Pi_D^{-1}(\Pi_C(\frac{m}{N})) = N * \frac{\frac{m}{N} - \alpha}{\beta - \alpha}$$

Let :

$$r = n' - n = N * \Pi_D^{-1}(\Pi_C(\frac{m}{N})) - n = N * \frac{\frac{m}{N} - \alpha}{\beta - \alpha} - n$$

Then, we know that:

if $r > 0$ **then**

the player will play strategy S_1 in the next generation.

else if $r < 0$ **then**

the player will play strategy S_2 in the next generation.

else if $r = 0$ **then**

the player will randomly choose its strategy in the next generation.

end if

The above analysis shows that r's value will be decided on whether the player will cooperate or defect. The critical value is when $r = 0$. We call the function when $r = 0$ the invasion function, which is:

$$N * \frac{\frac{m}{N} - \alpha}{\beta - \alpha} - n = 0, 0 \leq m, n \leq N$$

It can be rewritten as follows:

$$m = n\beta + (N - n)\alpha, 0 \leq m, n \leq N$$

This formula shows the number of cooperative neighbours needed for a cooperator to invade a defector which has n cooperative neighbours. When N is finite, the solution of the invasion function is a finite set of linear combinations of α and β, which we term the "invasion levels". The smallest value possible for the invasion level is when the strategy S_2 (defector) has no neighbouring strategies S_1 (cooperator), which is when $n = 0$. So, the invasion function will become $\frac{m}{N} = \alpha$, which is just the invasion point's function.

The escape point means that even when the cooperator has N cooperative neighbours, it still can't invade the defectors. So, for the escape point $m = N$. Then, the invasion function is $\frac{n}{N} = \frac{1-\alpha}{\beta-\alpha}$, which is as same as the escape point's function. For social dilemma games, the invasion functions are a series of monotonically decreasing linear functions defined on the 2D space of α and β. We posit that for an evolutionary game, if the neighbourhood for each player has a finite number of players, the results are predictable using the invasion function.

The cooperation rate of the evolutionary game will vary according to the values of m and n, starting from n = invade point of a cooperator (how many cooperative neighbours that a defector needs to invade a cooperator with no cooperative neighbours, which is smaller than 0 in the prisoner's dilemma, so we set $n = 0$), m = invade point of defector (how many cooperative neighbours that a cooperator needs to invade a defector with no cooperative neighbours, $\frac{m}{N} = \alpha$ in prisoner's dilemma), and ending at n = escape point of defector (how many cooperative neighbours that a defector needed so that even a cooperator with all cooperative neighbour can not invade it, $\frac{n}{N} = \frac{1-\alpha}{\beta-\alpha}$), and m = escape point of a cooperator (how many cooperative neighbours that a cooperator needed so that even a defector with all cooperative neighbour can not invade it, which is greater than N in prisoner's dilemma).

The cooperation of the evolutionary games will only be changed when α and β satisfies the invasion function, which is:

$$\frac{m}{\beta n + \alpha(N-n)} = 1$$

e call the formula of m and n which has pair of α and β that satisfy the above function an invasion level. So, the cooperation of the evolutionary games will only be changed on the invasion levels.

In other words, the prediction function defines a series of functions about parameters m, n, α, and β. However, we know that m and n represents the number of cooperative neighbours of the cooperator and defector, so they are integer numbers between 0 and N. Then we can get $N \times N$ functions, which are subset of the prediction function with integer m and n between 0 and N, that is the invasion level's function.

3.3 Experiments

To test the above assumptions, experiments have been undertaken on an 80×80 spatial graph. Each player plays a general two strategy game with all its neighbours in a Moore neighbourhood. To ease analysis, we initialise all games with the same initial generation; we

set one strategy to be that of a defector with the remainder being non-cooperative strategies. For subsequent generations, every player adopts the strategy of its neighbour who had the best payoff at the end of the previous generation. All players' strategies are updated in a synchronous manner.

We ran a number of experiments for varying game parameter values. We use a normalised game, and have tested the parameters $0 \leq \alpha < 1$, $1 < \beta \leq 2$ in increments of 0.01 resulting in 100×100 different parameter settings in total.

3.3.1 The prediction

Nowak [79] found that in the prisoner's dilemma ($\alpha = 0, \beta = b$), there exists transition points in the parameter space at which the evolutionary patterns fall into several category types; we believe these transition points are predictable, and the transition points are all defined on the invasion functions (a set of linear functions) that is the invasion level (a subset of the invasion functions).

The transition points can also be calculated from the invasion point and the escape point. We know invasion only happens when the cooperator has more than $N * I'$ cooperative neighbours and the defector has less than $N * E'$ cooperative neighbours (N is the number of neighbours in the neighbourhood). It is easy to know that with certain values of m and n, we can change the value of α and β to make the cooperators and the defectors have the same payoff. The invasion levels, or, the potential transition points, are defined by the combination of m and n, which is a linear function:

$$\frac{m}{\beta n + \alpha(N-n)} = 1$$

We suppose that between two invasion levels, a certain initial generation will always produce exactly the evolutionary.

Although there are 8×8 possible combinations of the possible values of m and n, however, due to the feature of the payoff of the game and the clustering of the graph, there are some restrictions of the value that m and n can take together.

We know that for the prisoner's dilemma, we have $1 \leq \beta < 2, 0 \leq \alpha < 1$, due to the prediction function $m = n\beta + (N-n)\alpha$, which is $\alpha = \frac{m}{8-n} - \frac{n}{8-n}\beta$ when $N = 8$. We can find that there must be $n < m < 3n$. So the possible of value that n can take with a m when N is:

1. $m = 2, n = 1$

2. $m = 3, n = 2$
3. $m = 4, n = 2; 3$
4. $m = 5, n = 3; 4$
5. $m = 6, n = 3; 4; 5$
6. $m = 7, n = 3; 4; 5; 6$
7. $m = 8, n = 3; 4; 5; 6; 7$ [1]

neighbourhood

So, we predict that in a lattice grid with the prisoner's dilemma, we have 18 invasion levels in total.

3.3.2 Experiment results

We recorded the different patterns (as Fig. 3.8 shows) that are generated by the evolutionary process with different game parameter settings using the same neighbourhood definition and the same initial configuration.

We depict the same evolutionary outcomes with one colour and depict the outcomes of all the 10000 experimental runs in to a 2D plot with β as x axis, and α as y axis. The plot is shown in Figure 3.7.

Figure 3.7 shows the distribution of the different patterns illustrating the emergence of cooperation in the game in an evolutionary spatial graph with Moore neighbourhoods. There are 7 different classes of patterns that emerge. It is worth noting that the grey colour does not represent one unique evolutionary outcome; however, all the evolutionary outcomes in the grey pattern all represent evolutionary runs with very quick convergence to full cooperation. The differences between these runs are not significant so we assign them to the same category.

The black points on the border represent those evolutionary patterns where the outcome is based on a random selection. This is because we define in our runs that when $r = 0$, the player will randomly adopt its strategy for the next generation. So, at the border of two patterns, there may be such points. Furthermore, as the value is not defined in a continuous space, the situation $r = 0$ may not exist on all points on the border.

[1] as Fig 3.6 shows, the two player playing aginest each other might not directelly linked, so m is possible to be 8.

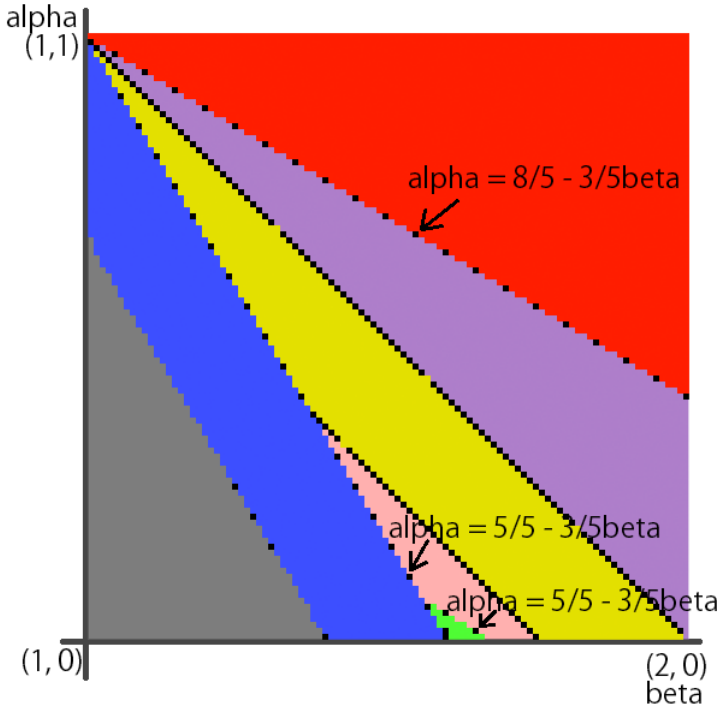

Fig. 3.7 The emergence of different patterns over different value of β and α in the pure strategy, 2-player prisoner's dilemma game. Each line represents the prediction function with different values for m and n.

Figure 3.7 shows that there are 7 observable evolutionary outcomes over all parameters settings. The grey pattern represents those outcomes with a convergence to cooperation while the red pattern represents those outcomes where defectors spread completely until no cooperation is left. The patterns are exactly separated according the invasion level; for example, the last invasion level's function (between the purple and red pattern) is $\alpha = \frac{8}{5} - \frac{3}{5}\beta$, which satisfies the invasion function which $m = 8$ and $n = 5$. It also satisfies the escape point's function which $n = 5$ (because $m = 8 = N$), which means for a defector with 5 cooperative neighbours can escape from any invasion of the cooperators. The red pattern is the last invasion level, or the escape level for the cooperators. The reason that the escape level is not at $n = 1$ is because the cluster coefficient is not zero in the Moore neighbourhood. So, we know that the cooperation of the experiments follows the prediction function.

However, not all the possible invasion levels appear in this experiment. It is because our initialization of the game is with only one defector in a group of cooperators. Some of the potential patterns may not show in the graph at all. Furthermore, we have also tried seeding the population with a different initial population and each of these experiments resulted in only a subset of the potential invasion levels. The most important thing is that we know that even if the initialization is random and the graph size is big enough to allow all the different pattern happen together, all of the different evolution results will still be predictable.

For example, the following evolutionary patterns are selected from one of the seven observable evolutionary classes of outcomes that arise; these are all located in the area of the graph coloured in green in Figure 3.7. The six pictures of the dynamic patterns show the distributions of co-operators and defectors in different generations during the evolution between the invasion level $I1$ ($\alpha = 5/5 - 3/5\beta$) and $I2$ ($\alpha = 8/3 - 5/3\beta$). The plot shows some of the history of the emergence of cooperation in this evolutionary pattern. All of the different combinations of the values of parameter α and β corresponding to the green area share the same evolutionary pattern.

The experiment results show that all the pattern changes are related to the changes on one of the invasion levels, and the cooperation rate increase as $\frac{m}{n}$ decreases. We can predict for all the possible pairs of the values of α and β that can influence the evolutionary results in all initializations. We also see that some of the invasion levels may not appear as the initialization may not give rise to all possible initial patterns of two neighbourhoods that can invade each other. However, the value of α and β that can change the evolutionary result in those games are still a sub-set of our predictions.

During the experiments, we realized that some invasion levels had been partially covered by others during the evolution. For example, for the blue, green and pink patterns, we

can see that the invasion level ($I1$) between the pink and green, with formula $\alpha = 5/5 - 3/5\beta, (m = 5, n = 3)$, intersects with another invasion level ($I2$) between the blue and pink, with formula $\alpha = 8/3 - 5/3\beta, (m = 8, n = 5)$. However, we realise that as long as the value of α and β are smaller than $I2$, then the results is in the blue pattern. This may because in the lattice grid, those two situations ($m = 5, n = 3$ and $m = 8, n = 5$) often occur together, which means, a player has a cooperative neighbour with 5 cooperative neighbours, and a defective neighbour with 3 cooperative neighbours, but at the same time, it has another cooperative neighbour with 8 cooperative neighbours, and another defecting neighbour with 5 cooperative neighbours, then, as long as the value of α and β do not satisfy the invasion level $I2$, the player will not change it strategy from cooperate to defect even if the first pair of cooperative neighbours satisfied the invasion level $I1$.

With the current model in this research, based on the possible number of cooperators of a cooperator to which a player is connected and the possible number of cooperators of a defector that the player is connected to, we can predict the effect of the game's payoff on regular graphs by examining the neighbourhood type of the graph. Although this prediction can predict for all the possible value of α and β may have influence the evolution, certain initializations of the population may still affect on the appearance of some invasion levels.

3.3.3 The definition of different games

From the result, we can also introduce new definitions of different games.

Definition 1 *The game can be looked as a linear function on the prediction map over the two parameters in the payoff matrix α and β.*

We can look at the 2-player's prisoner's dilemma, which has the payoff function, following normalisation.

$$P_{PD} = \begin{pmatrix} R & S \\ T & P \end{pmatrix}$$

and

$$P_{PD} = \begin{pmatrix} 1 & 0 \\ \beta & 0 \end{pmatrix}$$

So, this game can be defined as $\alpha = 0$ game, $(-1 < \alpha < 1, 1 < \beta < 2)$.

Considering the two-player snowdrift game (or the hawk and dove game):

$$P_{SD} = \begin{pmatrix} (b-c)/2 & b/2-c \\ b/2 & 0 \end{pmatrix}$$

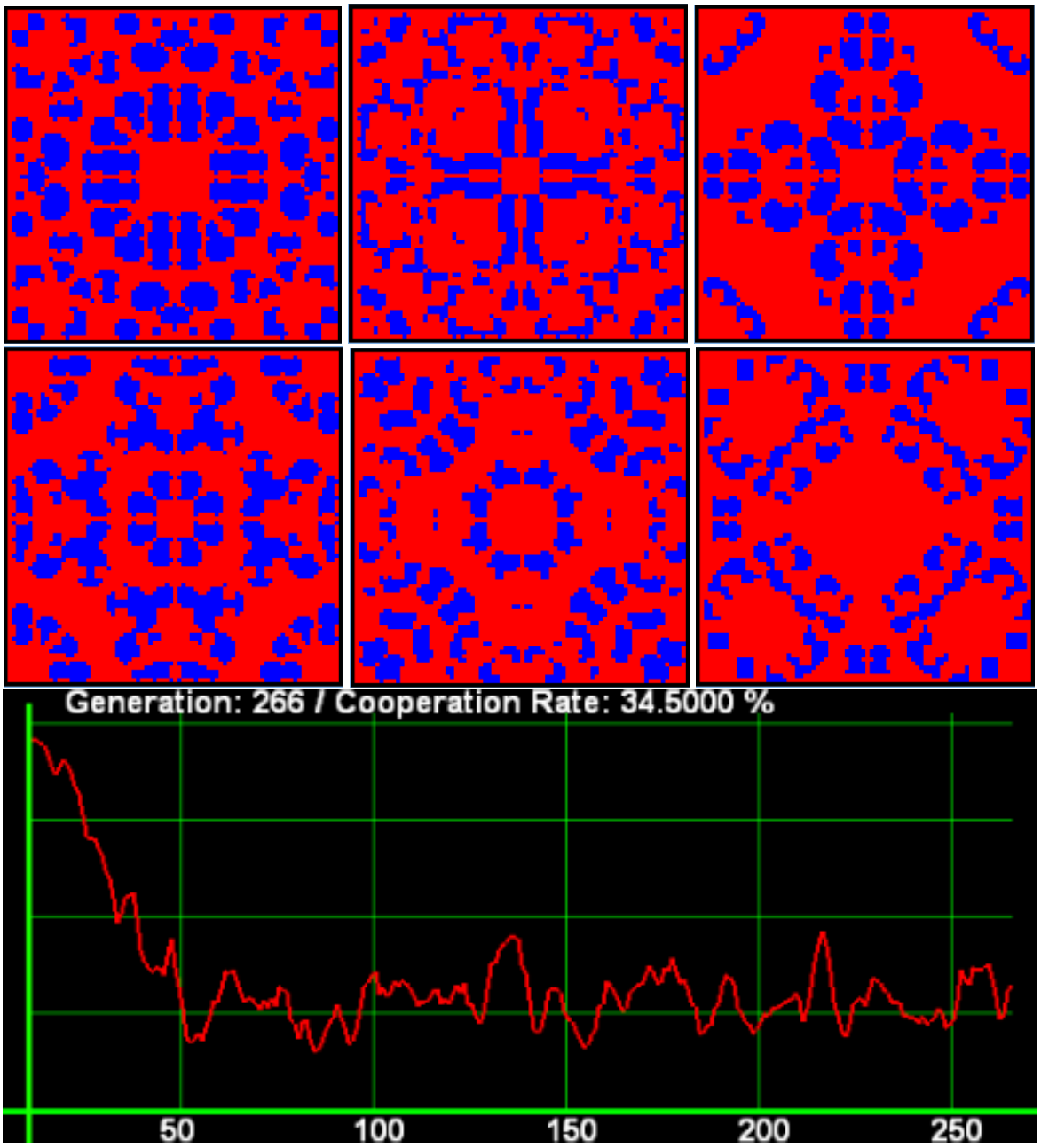

Fig. 3.8 For example, when the value of α and β is undertaken in the green invasion level, the evolution is highly unstable; the cooperation rate over generation is plotted.

which is, following normalisation:

$$P_{SD} = \begin{pmatrix} 1 & 0 \\ 1+\gamma & \gamma-1 \end{pmatrix}$$

It is an $\alpha = \beta - 2$ game, $(-1 < \alpha < 1,\ 1 < \beta < 2)$.

The emergence of cooperation in all games that have similar attributes (two strategy, linear payoffs) can be predicted with our approach.

By analysis of the neighbourhood, the prediction function can help us draw a map similar to Figure 3.7 and we can easily understand the different games' behaviours from this map.

3.4 Summary

This chapter concentrated on two strategy evolutionary games. A general model of the two strategy games has been proposed and analysed. From the payoff function of the normalized model, two points, the invasion and escape points have been considered to be important for the emergence of cooperation. A function has been defined using the invasion and escape points (termed the invasion function); it is defined as a set of lines on the 2D space that is defined by the two parameters of the normalized payoff matrix. These lines represent the invasion level in the evolutionary game. We proposed that with a fixed initial generation, the emergent pattern of cooperation only changes at the invasion levels. This fact can be used to predict the emergence of the cooperation of any two strategy game.

The experimental results show that for a 2 strategy, linear payoff game, we can develop a prediction function related to the type of its neighbourhood if it is played in a regular graph. From the prediction function, we get a series of invasion levels which relate to the range of the values of its payoff. Furthermore, we know that the emergence of cooperation of the game will only be influenced when the value of the game's payoff satisfies the function of the invasion levels.

Moreover, both the initial generation configuration and the neighbourhood may have an effect on the final result as well. However, the experimental results do show that all the actual invasion levels are one of the potential invasion levels that we can calculate from the invasion function. However, if the population is big enough and the initialization is random enough, all of the potential invasion level can be shown in the result, and the evolution of the game will uniform between each pair of invasion levels.

We have shown that the two-strategy games on spatial graph can be unified into a single

mathematical model which can be used to give a better understanding of the evolutionary patterns emerging in the simulation.

The main contribution of this chapter is that we discussed the effect of the neighbourhood in the prediction function, which gives us an approach to predict the emergence of the cooperation in the lattice game by calculating the invasion level.

In the later chapters, we'll include more strategies (iterated strategies), and complex networks.

Chapter 4

The Iterated Strategies

4.1 Iterated Games

In the last chapter, we analysed the evolution of cooperation in scenarios including two-player social dilemma games, with a strategy set size 2 on spatial graphs. We have proposed a prediction function to predict the potential evolutionary pattern during the evolution, based on the payoff matrix and the game parameters. In this chapter, we define a N-player game and a strategy set of size n for the IPD (iterated prisoner's dilemma), and we try explore the evolution of cooperation with additional iterated strategies.

Given the success of the TFT (Tit for Tat) strategy in the two-player games [5, 6], in this chapter, we decide to explore the emergence of cooperation in a population of generalised TFT strategies, which we called N-player TFT (or TFTn for short). The main objective is to examine the behaviour of the iterated strategies evolving on a lattice grid, and, if the society can be more cooperative by incorporating the TFTn strategies. We also compare the different "tolerance" levels of the TFTn strategies, and found that lower tolerance can provide increased evolutionary stability against defectors.

4.2 N player prisoner dilemmas

The traditional TFT strategy is defined for two-player games, and each player starts as a cooperator, and updates its move based on the opponent's move in the previous game. However, a N-player version can be defined where the player plays with $N-1$ opponents together at the same time [19]. There are two possible approaches:

1. The player plays a 2-player game with each of its opponents independently, and the

player receives the average payoff of the $N-1$ games. The classic payoff function can be adopted.

2. All of the N players (includes the player itself) play one game together, and the player receives the pay-off of this N-player game. A payoff function must be designed for this case.

For the traditional evolutionary game where the strategy set only includes pure C and pure D, both of the two approaches above return the same payoff score to the players, e.g. for a player that has m cooperator and n defector neighbours, will receive a payoff of $m \times R + n \times S$ by playing N two-players games, and playing one N players game, the payoff of the same player will be $(m \times R + n \times S)/(m+n) - cost$. As $(m+n)$ and cost are constant, the payoff are exactly the same.

However, for an iterated game which includes strategies such as TFT, the payoffs received by each player can differ under these different conditions. It is because the TFT strategy decides its next move based on its opponent's previous move. In playing N separate two-player games, the TFT player can adopt different moves against different opponents. In a N-player game, player can only adopt a move based on all of its opponents' last moves. In effect, in the two-player game, a strategy can punish or reciprocate cooperation of individual players whereas in the N player game a strategy must reward or punish the group as a whole.

The traditional 2-player TFT strategy is mainly determined by following conditions:

1. Always cooperate on the first move.
2. If the opponent defects, the TFT player will retaliate with a defection.
3. TFT player is willing to forgive, i.e. a cooperative move by the opponent will be followed with a cooperation.

In this research, we use a set of N-player TFT strategies. Suppose that a TFT player is playing against N opponents in an iterative N-player game. In each iteration, the player looks all of his opponents' moves in the previous turn, and then decides the next move based on the number of cooperative opponents.

N-player TFT strategy Definition 1 *The N-player TFT strategy (TFTn for short), always cooperates at the first move, and moves as defect in an iteration when it has less than n ($0 < n \leq N$) cooperative opponents in the previous iteration, and cooperates in all other*

cases. We call n the level of tolerance. When n is small, we say the level of tolerance is higher, as it can tolerate more defecting opponents.

In this research, we ran the evolutionary iterated game on a lattice graph with a Moore neighbourhood (with 8 immediate neighbours). We have 10 N-player TFTn strategies, including TFT0 (pure C), TFT1 to TFT8, and TFT9. The pure-C strategy can be viewed as TFT0, as it always starts with a cooperation and does not need cooperative neighbours to continue cooperating. It is also worth noting the pure-D strategy (always defect) is not equivalent to TFT9 as pure-D always starts with a defection, whereas TFT9 will cooperate on the first move before defecting for the remainder of the game.

4.3 Experiments and results

In these experiments, the players are playing N-player Prisoner's Dilemma [133] on an 81×81 lattice graph [1]. With adaptable parameters of the game's pay-off, we compared the experimental results of the non-iterated game and the iterated game using N-player TFT strategies. The experiments are first run on a lattice graph of players with randomly initialized strategies; this is mainly to observe the overall cooperation rate of the final state (this may or may not be an evolutionary stable state) of the game. Certain pre-designed initial patterns are also considered for the rest of the experiments, to study the behaviour of the TFT on the iterated evolutionary game. The overall cooperation rate is decided by the final move [2] of each generation. For example, if the final move of a TFT strategy is C, then we count it as C when calculating the cooperation rate. This is because we only want to know how many players are cooperating; in these experiments, as the TFT strategies will always start with cooperation, if it ends as a cooperator in the final turn in one generation, it definitely cooperates at the next generation, so it doesn't matter if the strategy is TFT or pure C if the final move is cooperate, since they will make the same contribution to the society in the next generation.

4.3.1 Iterated games on random initialized lattice graphs

In the first experiment, we randomly assign the strategy for each player at the first generation, and then start the game. In each generation, each player will play with its 8 immediate

[1] As the cooperation is mainly depends on the neighbourhood, the dimension can be vary to any size

[2] a move is a step of strategy change followed the TFTn rules in one generation, there are 10 moves in each generation, which decided by the number of iterated rounds for the TFTn strategy. this number can be vary.

neighbours, and then evolve to the next generation, with each player learning from its neighbour with the best pay-off, at the beginning of the generation. The overall cooperation rate is calculated at the end of the generation.

Experiments setting

The experiments started with a N-player prisoner's dilemma game on a lattice grid, with the parameters set to: $R(Reward) = 1.0, S(Sucker) = 0.0, T(Temptation) = \beta, P(Punishment) = 0.0$. We compare the non-iterated game with only the pure C and pure D strategies present and the iterated game which also has the TFT4 strategy in addition to the pure C and pure D strategy. The iteration number i, is set to 10 as default. To minimise the influence of the random initialization, each experiment has been run 200 independent times to obtain the average value. The experiments show that the majority of the TFT4 players will end up playing a cooperate strategy on the last move of the generation. In other words, those TFT4 player who are defecting in the previous generation, will have higher probability to be invaded by defectors during the evolution.

The research in this chapter is aimed at analysing how different strategies can affect the cooperation, so we want to minimise the effect of the game's payoff. Through the analysis in the last chapter, in this experiment, to demonstrate the ability of each TFTn strategy, we decide to run experiments with $\beta = 1.51$ and $\beta = 1.61$ respectively under the same circumstances. 1.51 and 1.61 are intermediate values for β that do not significantly benefit either cooperators or defectors.

Experiments with $\beta = 1.51$

In the experiments with $\beta = 1.51$ [3] (plot 4.1), we found that although the iterated game has more cooperators at the first generation (because the initial generation is randomly generated with 3 strategies: pure C, pure D, and TFT4, the TFT4 will always start as a cooperator, so there will be approximately 66% of cooperators rather than 50% in the non-iterated game), but the cooperation rate (for both cooperators and the TFT players who ended up to cooperate) of its final state actually decreased to around 82% from the original 88%, the significance can be clearly observed from Plot 4.1, which means having a few TFT4 players in the society does not promote the cooperation rate and in fact, may even decrease it. However, the iterated game still significantly increased the speed at which the society evolved to a relatively stable state. We have also tested the game which only includes pure defectors

[3] 1.51 is used to avoid that cooperator may have same payoff with defector.

and TFT4 with the same parameter set; the cooperation rate of its final state is around 80%, the cooperation rate of the 1st generation of the game is very low, it is because we measure the cooperation rate at the end of each generation (after the iteration), so, most TFT players that are connected to a lots of defectors have adopted defection, however, the cooperation rate is increasing as the TFT player has conquered some entire places, and they will continue cooperating with each other to prevent the invasion of the defectors.

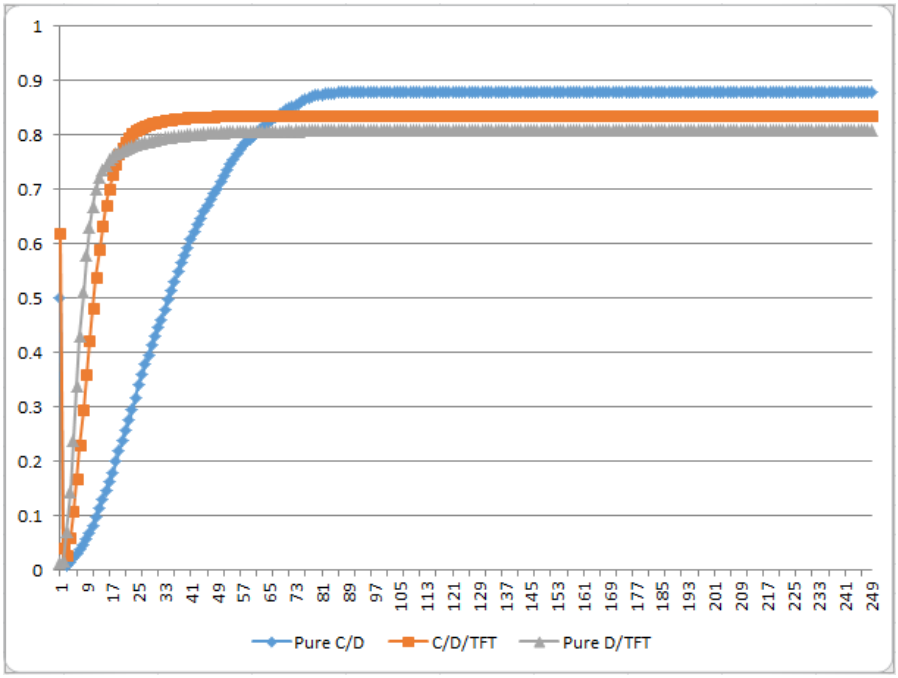

Fig. 4.1 Game with pure C/D strategy set, pure C/D, TFT4 strategy set, and pure D and TFT4 strategy set start with random initialization $\beta = 1.51$

Then, we changed the level of tolerance from 1 to 8, with and without pure C players (plot 4.2, and plot 4.3). It was surprising that the cooperation rate of the final state can be increased by either increasing or decreasing the level of tolerance. So, in the extreme situation (such as TFT1 or TFT8), the cooperation rate of the final state will be much higher.

Plot 4.2 and plot 4.3 are the plots of TFT1-TFT8 strategy players playing against pure C/D and pure D neighbours respectively in a random initialized generation with $\beta = 1.51$. We can see in both situations, adopting the TFT8 strategy (which is to defect as long as it played with 1 defector on the previous turn) has been shown to have the highest cooperation rate. For some reason which is unknown, TFT4 has the worst performance when $\beta = 1.51$.

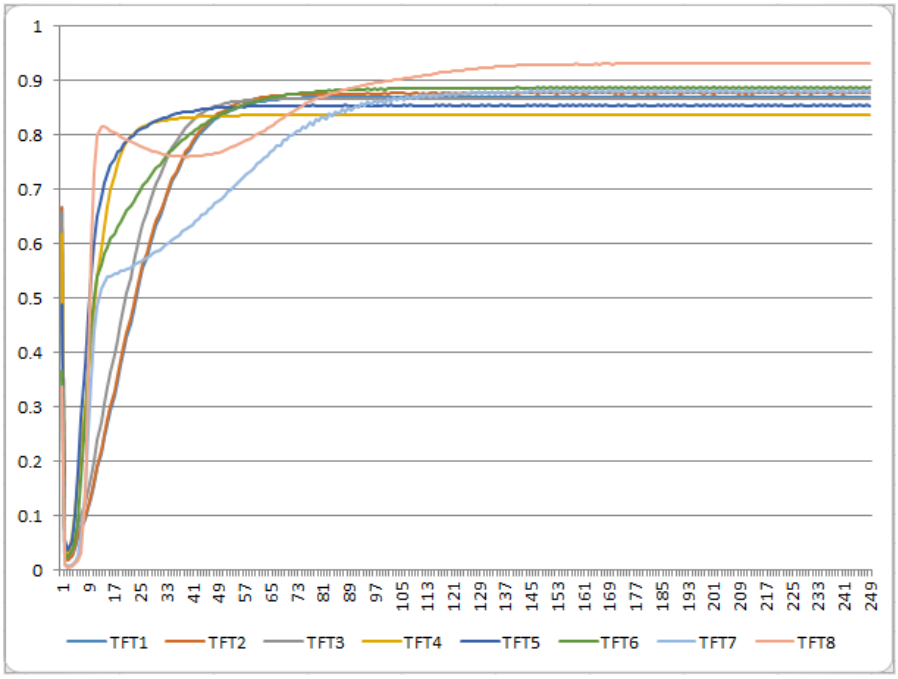

Fig. 4.2 The TFTN strategy player playing with Pure C/D strategy players start with random initialization $\beta = 1.51$

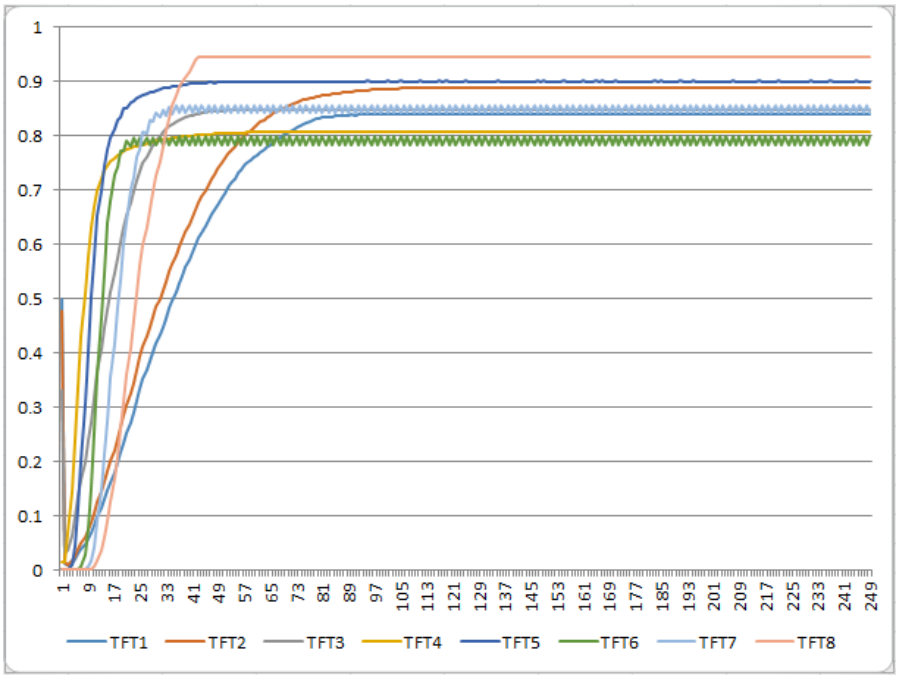

Fig. 4.3 The TFTN strategy player playing only with Pure D strategy players start with random initialization $\beta = 1.51$

Experiments with $\beta = 1.61$

As seen in Plot 4.2 and plot 4.3, the difference in the cooperation rate between different TFTn, is quite small; in order to fully explain the difference, we increase the value of β to 1.61, and run the experiments again. Plot 4.4 shows the pure C/D game, pureC/D and TFT4 game, and the Pure D and TFT4 game on random initialization with $\beta = 1.61$. It is easy to see that unlike the experiment results showed in Fig 4.1 which have the same set up except $\beta = 1.51$, the pure C strategy does not gain an advantage against the pure D strategy any more (the experiments that have only pure C and pure D strategies have a much lower cooperation rate in comparison with other setups). By contrast, the TFT4 strategy can maintain a relatively higher cooperation rate in both games that have only pure D players as opponents and have both pure C and pure D players as opponents.

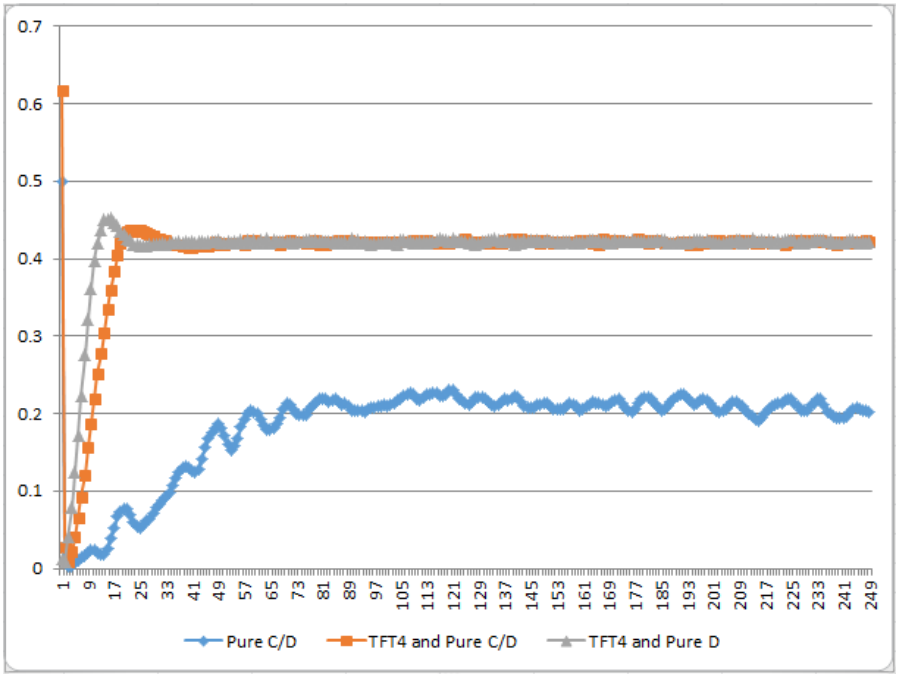

Fig. 4.4 Game with pure C/D strategy set, pure C/D, TFT4 strategy set, and pure D and TFT4 strategy set start with random initialization $\beta = 1.61$

Plot 4.5 and plot 4.6 are the plots of TFT1-TFT8 strategy players playing against pure C/D and pure D neighbours respectively in a random initialized generation with $\beta = 1.61$.

From both plots 4.5 and plot 4.6, we found that the final cooperation rate is increasing while the level of tolerance is decreasing, the highest cooperative society is the TFT8, which is the lowest level of tolerance. In other words, the more TFT will tolerate defectors, the

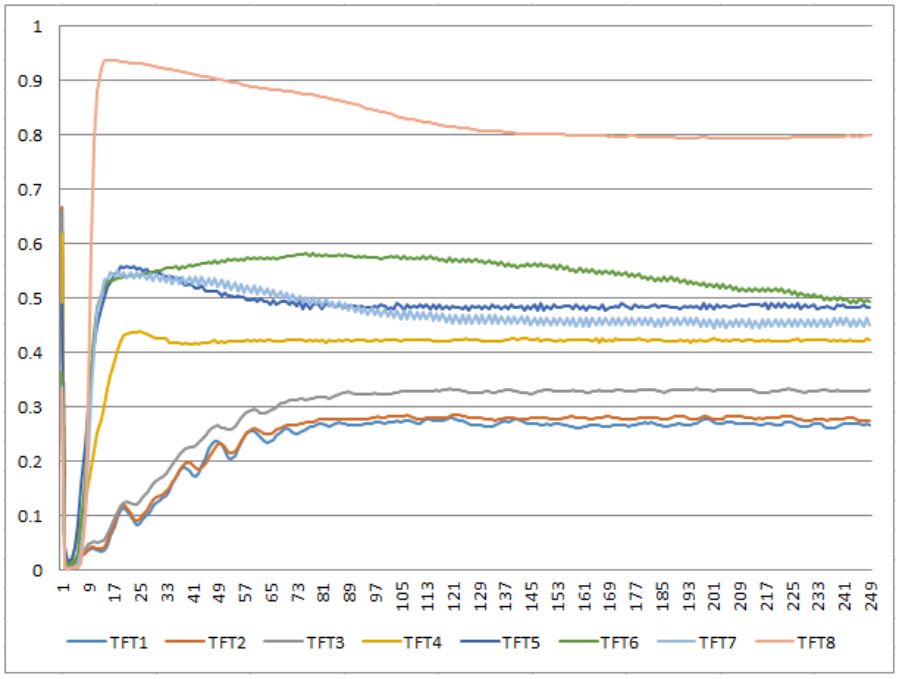

Fig. 4.5 The TFTN strategy player playing with Pure C/D strategy players start with random initialization $\beta = 1.61$

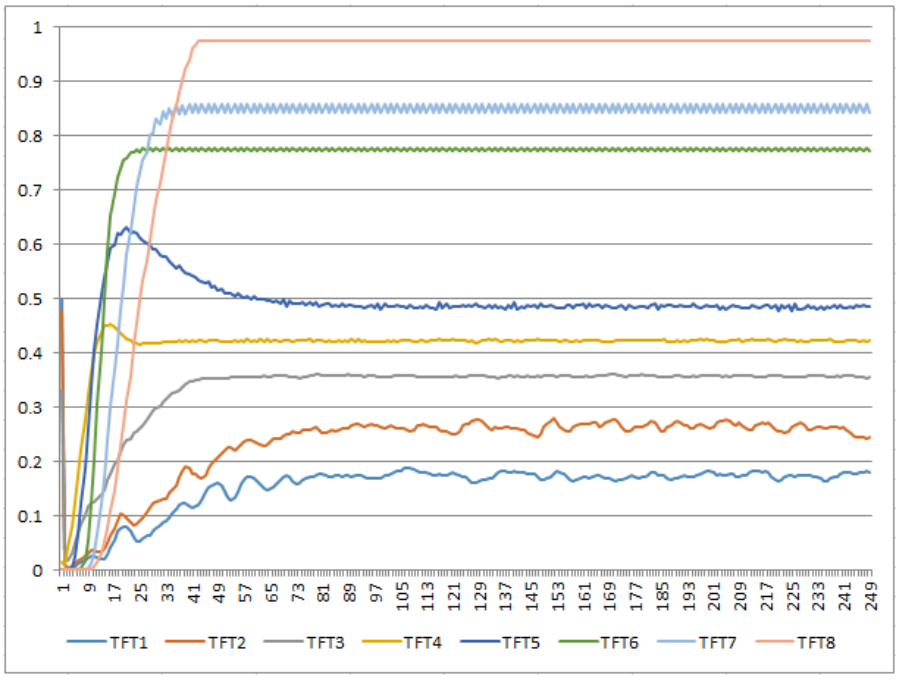

Fig. 4.6 The TFTN strategy player playing only with Pure D strategy players start with random initialization $\beta = 1.61$

easier it to be invaded by the defect strategy. Also, notice that the TFT4 no longer has the "unexpected" performance witnessed when $b = 1.51$.

There is also another interesting phenomenon in that when the level of tolerance is very low, such as TFT8, there is always a big oscillation in the cooperation rate in the earlier generations. The cooperation rate in the randomly initialized evolutionary game usually decreases to a very low level due to the defector taking advantage of neighbouring strategies on the random allocation, and then increasing to the final state, the TFT8 will have a big decrease in cooperation rate in the middle, and then slowly increase to the final state.

Comparison over different values of β

Finally, we ran an experiment with initially pure C, pure D, and the TFTn strategy players randomly mixed together. Two experiments have been run with temptation payoff 1.51 and 1.61 respectively. The results are depicted plot 4.7. The TFT players always start as cooperators, there are only about 10% players playing defect at the first generation. Due to the fact that the number of players playing the defect strategy at the first generation may influence the final result, we increased the number of defectors to make sure that around $1/3$ of the players in the initial population are pure D player, and then randomised the remaining $2/3$ of population to play the other 9 strategies. Then, we obtained results depicted in plot 4.8, to our surprise, the experimental result is quite similar to plot 4.7. The cooperation rate of the final state of this experiment is very similar to the result which only contained TFT5 and pureD players, and the lower tolerance TFT players (such as TFT5,6,7,8) are more likely to survive under the invasion of the defectors than the pure C and the higher tolerance TFT players.

4.3.2 Iterated games on pre-set patterns

The random initialization adds much variation in the evolution, it gives us an insight into the robustness of strategies against each other, but it gives no intuition regarding the behaviour of a particular strategy. For this reason, we used pre designed initial patterns with fixed attributes in the experiments, which are more understandable during the analysis of the behaviour of the iterated strategies.

One of the most commonly used pre-set pattern is the Nowak and May's pre-set (NM pre-set), defined by Nowak and May [81], which is to allocate one defector in a group of cooperators. As the TFT player always performs cooperatively at the beginning of the iteration, the experiment has been set as one defector in a group of TFT players. Also, extra

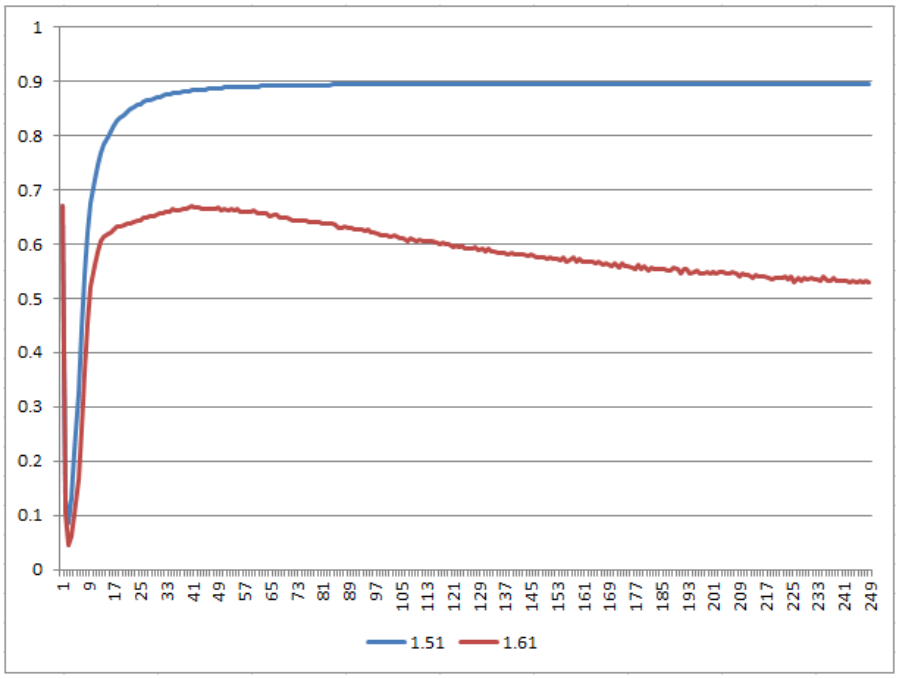

Fig. 4.7 Pure C/D and TFTN strategy randomly mixed, $\beta = 1.51$ / $\beta = 1.61$

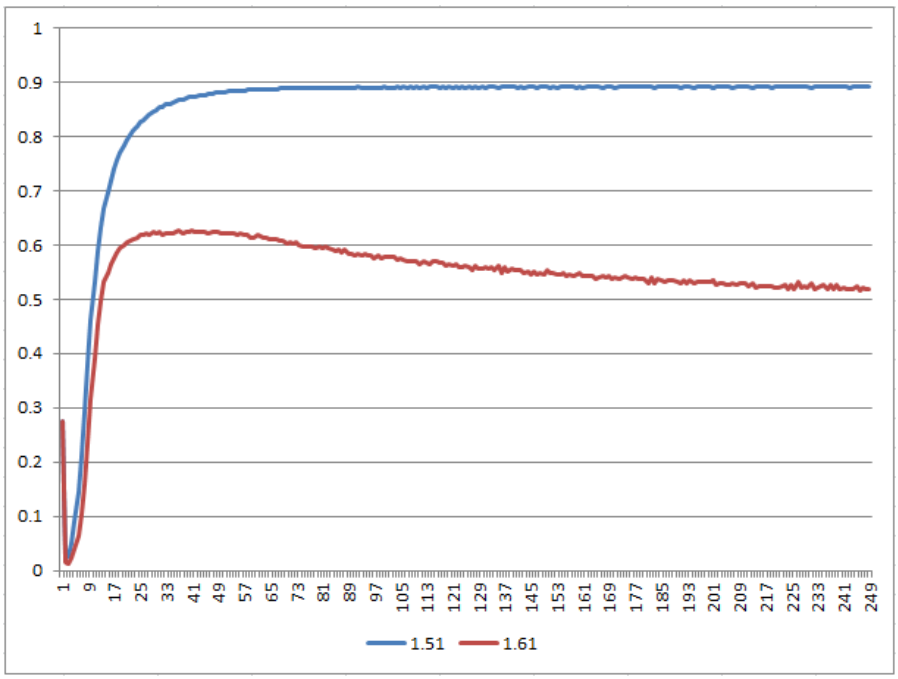

Fig. 4.8 Pure C/D and TFTN strategy mixed, has $1/3$ pure D at the first generation. $\beta = 1.51$ / $\beta = 1.61$

experiments have been ran with 3×3 cooperators in a group of defectors and 3×3 TFT players in a group of defectors.

As was shown by Nowak and May [79–82], in the NM pre-set, the evolution is highly dependent on the temptation pay-off (which is represented as β here), and some times, it may show different evolutionary patterns with a certain setting of β. Plot 4.9 is the experiment results of a defector's invasion in a group of TFT0(pure C) to TFT8 players when $\beta = 1.61$.

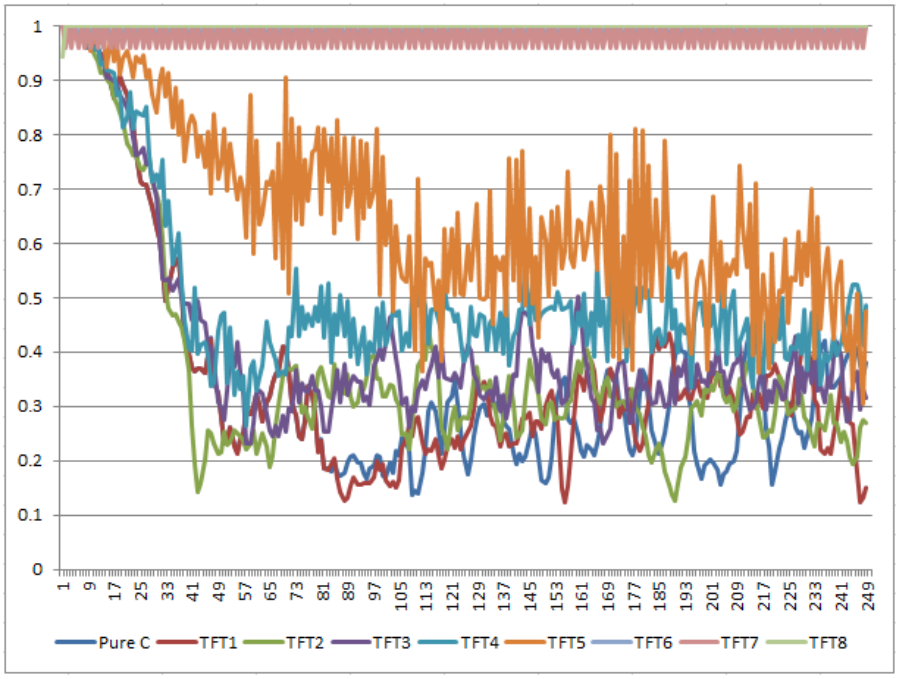

Fig. 4.9 One defector's invasion in a group of TFT0 (pure C) to TFT8 players $\beta = 1.61$

The experiment results showed that for defectors that have been allocated in a group of cooperators, it is much easier to invade others comparing with most defectors that have been allocated in a group of TFT players. Moreover, the lower the level of tolerance, the harder it is for the defectors to invade the TFT players. When $\beta = 1.61$, it is similar with the random initialization experiments above. We have also tried other different initial patterns (such as 5 defectors in a group of TFT players), and the results are very similar.

4.4 Summary

In this chapter, we proposed and ran the evolutionary game with an iterated-strategy (TFT). By introducing a set of N players TFT strategies, and comparing them to the non-iterated

strategies in experiments with both randomly initialized generation and pre-set initial generation, we found that although in comparison with a pure C strategy, an iterated strategy such as TFT, has a better chance of invading the pure D strategy. However, it does not guarantee that bringing TFT players into a society including both pure C and pure D players always increases the final cooperation rate. Actually, it may even decrease the cooperation rate with certain setting of the game (Random initialization, $\beta = 1.51$). We also found that, sometimes, for the N-player TFT, when the value of the temptation payoff β is low, either increasing or decreasing the level of tolerance can promote the cooperation. But this situation will no longer happens when the value of β is high (e.g., when β is raised to 1.61), the level of cooperation increases as the level of tolerance decreases. For most times, although the lower level of tolerance (TFT6, 7, and 8) may add more oscillation to the level of cooperation during the evolution, it can always maintain a high level of cooperation during the entire evolution. This is found in all experiments especially in the experiment with presetted initializations when $\beta = 1.61$. Over all, the lowest level of tolerance strategy, which is the TFT8 strategy, has the best performance against the defectors. In all of the experiments that we have ran, the TFT8 strategy can keep a relatively higher level of cooperation and outperforms all other strategies. Interestingly, higher level of tolerance will decrease the cooperation rate which is not as we expected.

In chapter 3 and 4, we examined the emergence of cooperation on the spatial graph (lattice grid specifically), with 2-player and N-player social dilemma game separately. In the further chapters, we'll adopt complex graph topologies instead of the lattice topology. As the iterated games can not guarantee a promote on the emergence of cooperation, we decide to adopt only pure C and D player with a payoff matrix $P = \begin{pmatrix} 1.0 & 0.0 \\ \beta & 0.0 \end{pmatrix}$ to minimize the effect of the game's parameter on the evolution of cooperation, where β could be set to 1.51 or 1.61 depends on the situation. Also, with fixed the game's parameter, we could concentrate more on the study of the structure of graph.

Chapter 5

Graph Topology and The Robustness of Cooperation

5.1 From Regular to Complex Graph

Unlike regular graphs, the evolutionary dynamics are hard to predict in more complex graphs. Also, a complex graph may contain many attributes which may influence the outcome; this can be hard to analyse mathematically. A common approach to study the evolutionary dynamics is through simulation. Many simulations showed that heterogeneous graphs are more robust regarding cooperation in comparison to regular graphs [95, 108, 109]; this is normally related to the scale-free property of the graph. However, there are no proofs that the power-law degree distribution is the only reason for scale-free networks' outstanding performance in maintaining cooperation. Much research has shown that the node to node correlation is another factor in a graph that may lead to highly robust cooperation [96].

There are common parameters in a graph that may have an impact on the emergence of cooperation, such as the average degree [36, 118] and the clustering coefficient [57, 131]. Tang and his colleagues discussed the role of connectivity (average degree) on the evolution of cooperation. Their experiments show that the cooperation rate has a maximum value within a certain average degree and follows a one-peak function [118]. Ren examined the evolutionary dynamics on small-world networks with different levels of randomness, and found that the randomness can promote cooperation in a resonance-like way [1] [101]. Wu et al. suggested that the resonance-like phenomena is actually related to the clustering

[1] The cooperation level has a tendency to oscillate with greater amplitude at some frequencies than at others.

coefficient instead of the small-world effect, as it is also exists in a regular ring graph, which does not exhibit the small-world effect [131]. Also, Lei et al. believe that the cooperation rate will only increase when the hubs are occupied by cooperators in a graph exhibiting a high clustering coefficient [57].

There has been some work in examining the role of the clustering coefficient [57, 131]. However, the main focus in those work is to model real world networks, and hence the clustering coefficient, number of nodes, edges and degree are not controlled. Although the experiments are sufficient to explore the emergence of cooperation on these graphs, they do not fully explain the effect of the clustering coefficient in isolation of other parameter's influence. We attempt to explore the effect of the node degree and clustering coefficient (transitivity) by fixing the other graph parameters as much as possible. We then compare the experimental results of the evolution of cooperation on a series of graphs with the same size (and the same number of edges if necessary), over a range of clustering coefficient values. Hopefully these results will help show the effect that the clustering coefficient has on the emergence of cooperation.

Our approach has three main steps:

1. Create a graph generation algorithm that can generate graphs that satisfy all the requirements for our experiments.

2. Create the graphs and analyse mathematically the relationship between the graph structure and the clustering coefficient and attempt to predict the outcomes of simulations using this analysis.

3. Run the simulation experiments and observe the results. Compare the observed results with our predictions.

5.2 Generation of graphs with adjustable clustering coefficient

The clustering coefficient can be calculated for a given node as the number of its neighbours that are connected over the total number of potential connections between them. Therefore, for the entire graph, the clustering coefficient can be calculated as the average of the clustering coefficient of each node (which is called the average clustering coefficient), or the number of triangles over the number of open triplets in the graph (also called transitivity). They both represent the graph's clustered architecture, with a little difference on the scaling

of the value. However, the most common approach is using the transitivity (or the global clustering coefficient).

Most of the previous approaches to generating graphs with a tunable clustering coefficient attempt to model real-world networks which have the following common properties [47]:

1. Skewed degree distribution.
2. Low mean geodesic distance.
3. High clustering coefficient.

Many of the existing graph generation algorithms with tunable clustering coefficients are based on the preferential attachment model (where a higher degree vertex has a higher probability of attaching to new neighbours [97]) to generate graphs exhibiting a power law in node degree [43, 47, 52]. However, those approaches are only suitable for power law graphs with relatively low clustering coefficients. Several graph generation algorithms [47] are inspired by the Newman algorithm [77], which is based on the configuration model [74]. Experiments show that the clustering coefficients of the graphs generated by those algorithms may have increasing errors as the average degree increases [47]. There are also other algorithms that use degree sequence [47] and random walks [48].

In attempting to build a graph that closely models real-world graphs, most of the algorithms mentioned above can not guarantee a connected graph with a minimum error on the clustering coefficient. In this work, in addition to generating graphs with a tunable clustering coefficient, we also need the generated graphs to satisfy the following conditions:

1. Both the clustering coefficient and the graph size can be controlled.
2. The precision of the clustering coefficient is controllable (within a pre-defined error).
3. All of the vertices of the graph should be connected, i.e., there are no isolated subgraphs.

Considering all of the above conditions, we propose 2 new graph generation algorithms based on the configuration model in this work.

Algorithm 1 Ascending graph generate algorithm

1. Create a connected graph (with a clustering coefficient = 0).
Create a graph with 3 vertices V_0, V_1, and V_2.
Add 2 edges to the graph, linking V_0 and V_1, and linking V_1 and V_2.
while $i < N$ **do**
    Add a new vertex V_i to the graph.
    Randomly link V_i to an existing vertex V_r, $(0 <= r < i)$.
end while
2. Keep adding edges (which can guarantee the addition of new triangles), until the actual clustering coefficient is close enough to the expected clustering coefficient.
while $pc - ac > err$ **do**
    Randomly pick a vertex V_n.
    if V_n has two unconnected neighbours V_p and V_q **then**
        connect V_p and V_q.
    end if
end while

5.2.1 Ascending graph generate algorithm (ascGen)

Given that we wish to generate a graph with N vertices and the desired total clustering coefficient is pc, let the actual clustering coefficient be denoted as ac, and the allowed error denoted as err, the pseudo-code of the ascGen algorithm is show in Algorithm 1

Note that there are some mechanisms adopted to prevent the algorithm from entering infinite loops.

The ascGen algorithm can generate the desired graph with a fixed number of nodes and a specified clustering coefficient in $O(V^3)$ time [2]. Moreover, to decrease the time spent on generating the graph, according to whether the desired transitivity is high or low, the graph can be constructed by taking a connected graph with clustering coefficient 0 and then adding edges to the graph; or from a complete graph and then removing edges form it. However, this approach has several drawbacks:

1. The number of edges in the graph increases until the desired clustering co-efficient is found. Hence, it is impossible to control the number of edges.

2. For graphs that have the same number of nodes, the clustering coefficient could be the same and yet have a different number of edges. As the edge that is added each time is not guaranteed to add new triangles (and hopefully increase the clustering coefficient), the graph that ascGen generates always has a relatively low number of edges (but does

[2] which is similar to the time complexity to calculate the clustering coefficient

not guarantee the minimum number of edges).

3. When the number of edges in the graph is low, each time we add an edge to create new triangles, the clustering coefficient increases, but when the number of edges is relative high in comparison to the number of nodes, each time we add a new edge, it is highly possible to add many more triplets, so, the clustering coefficient may decrease for a number of iterations as the edge number increases.

As we may create graphs with a desired clustering coefficient and a varying number of edges (and hence average degree), we cannot explore the effect of the clustering coefficient alone with these graphs. Another algorithm is required to create graphs of a fixed size, a specified clustering coefficient and a fixed number of edges.

5.2.2 Heuristic graph generation algorithm

We wish to generate a graph with N vertices, E edges, and a desired total clustering coefficient, pc. Let the actual number of edges be denoted as e, and let the actual clustering coefficient be denoted as ac, and let the allowed error be denoted as e, the pseudo code of the heuristic graph generation algorithm is presented as Algorithm 2

5.3 Graph property and the emergence of cooperation

5.3.1 Graph architecture

Given two graphs with the same number of nodes and edges, we may generate graphs with different clustering coefficients. Clustering coefficients can range from zero (e.g a tree structure) to one (a complete graph).

We adopt the heuristic graph generation algorithm to generate several graphs exhibiting the highest clustering coefficient with a constraint on both the number of vertices and the number of edges. The generated graphs with the maximum clustering coefficient had similarities in structure; the maximum clustering coefficient was achieved with graphs containing a set of complete sub graphs each of the same size where each of the subgraphs was connected to other subgraphs by an edge (a bridge). A sample graph is depicted in Figure 5.1.

For a complete graph with N vertices, the graph has $\frac{N\times(N-1)}{2}$ edges. The clustering coefficient of the graph will be 1 as we have the same number of triples as we do triangles

Algorithm 2 Heuristic graph generate algorithm

1. Create a fully connected graph (with clustering coefficient = 0)
Create a graph with 3 vertices V_0, V_1, and V_2.
Add 2 edges to the graph, linking V_0 to V_1, and linking V_1 to V_2.
while $i < N$ **do**
    Add a new vertex V_i to the graph.
    Randomly link V_i to an existing vertex V_r, $(0 <= r < i)$.
end while
2. Add the rest of the edges to the connected graph randomly to generate a connected random graph with expected size and degree
while $e < E$ **do**
    Add a new edge randomly
end while
3. Keep removing and adding new edge until the actual clustering coefficient is close enough to the expected clustering coefficient
while $pc - ac > err$ **do**
    for each edge e **do**
        if the edge is not the only path between its two vertices **then**
            Calculate the potential clustering coefficient if this edge is deleted
        end if
    end for
    Delete the edge which produces a new clustering coefficient closest to the expected clustering coefficient following deletion
    for each node i **do**
        for each node j **do**
            if Node i and Node j are not connected **then**
                Calculate the potential clustering coefficient if we connect Node i and Node j
            end if
        end for
    end for
    Connect the two nodes which produce a new clustering coefficient closest to the expected clustering coefficient following connection
end while

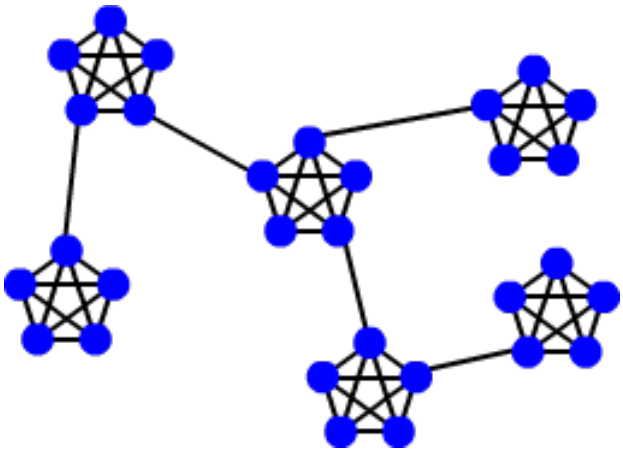

Fig. 5.1 The graph created by 6 complete sub graphs of size 5

[3]. By connecting two complete graphs each of size N, the new graph will have the number edges given by:

$$2 \times \frac{N \times (N-1)}{2} + 1 = N \times (N-1) + 1$$

The clustering coefficient can be calculated as:

$$\frac{\frac{N \times (N-1) \times (N-2)}{2}}{\frac{N \times (N-1) \times (N-2)}{2} + 2 \times (N-1)} = \frac{N^2 - 2N}{N^2 - 2N + 2}$$

Furthermore, for M complete sub graphs each with N vertices linked together by single bridges (no vertex has been chosen to be a point on more than one bridge), when $M \to \infty$, the clustering coefficient is:

$$\lim_{M \to \infty} \frac{M \times \frac{N \times (N-1) \times (N-2)}{2}}{M \times \frac{N \times (N-1) \times (N-2)}{2} + 2 \times (N-1)} = \frac{N^2 - 2N}{N^2 - 2N + 4}$$

Therefore, the maximum clustering coefficient of a graph with a fixed known number of both vertices and edges can be estimated. For example, to calculate the maximum clustering coefficient for a graph with n vertices and an average degree of d, we know the maximum clustering coefficient is when the graph comprises several complete sub graphs connected by bridging links. A complete graph with size s, has an average degree of $s-1$. If many of those graphs are connected together, the average degree will be roughly equal to $\frac{(s-2) \times (s-1) + 2 \times s}{s}$ (ignoring the nodes with the extra link connecting the subgraphs).

[3] Number of triangles is given by $C_N^3 \times 3 = \frac{N \times (N-1) \times (N-2)}{2}$

Therefore, the maximum clustering coefficient of a graph with n vertices and an average degree, d, is approximately the clustering coefficient of a graph comprising $m = n/([d]+1)$ complete subgraphs each with size $[d]+1$, which has the clustering coefficient of roughly $\frac{d^2-2d}{d^2-2d+4}$ when m is small, and roughly $\frac{d^2-2d}{d^2-2d+2}$, when m is big.

Conversely, if we wish to calculate the lowest possible clustering coefficient for a graph, we must consider a graph with the minimum number of triangles in the graph. We know the clustering coefficient of any graph is equal to 0, provided there are no triangles in the graph.

In terms of minimising the number of edges in a graph with a clustering coefficient equal to zero, the smallest possible structure of the graph is a rectangle. In order to build a graph with the maximum number of edges for a given fixed size of nodes, N, we can construct a regular graph constituted by rectangle structures. (rectangle structure is a graph structure in which the maximum distance between two nodes is 2). In other words, for a graph with size N, with a maximum number of the edges and a clustering coefficient = 0, the graph will be a regular graph with an average degree equal to $\lfloor N/2 \rfloor$.

5.3.2 The robustness of cooperation

The previous sections describe how different graphs with different clustering coefficients and a fixed number of vertices and edges can be created. By analysing the robustness of cooperation on these different graphs, we can systematically explore the clustering coefficient's influence on the emergence of cooperation.

The analysis below shows how the structure of the graph influences cooperation levels.

Let's assume the game to be run on the graph is a $N \times 2$ player game with the following payoff matrix: $\begin{bmatrix} 1 & 0 \\ \beta & \alpha \end{bmatrix}$

The players learn from their neighbour who receive the highest payoff. If the game is a social dilemma, then the following constraint holds: $\alpha < 1 < \beta < 2$.

For a defector with neighbours who are all cooperators, the average payoff received by the defector is $\beta > 1$, which means, it will obtain a greater score than its cooperative neighbours and will therefore invade its cooperative neighbours. As for any cooperator with at least one non-cooperative neighbour, the average payoff it receives is less than 1. Moreover, if the degree of the graph is sufficiently high, defection can spread and occupy the entire graph in a few generations, as Figure 5.2.

However, if a defector also has other neighbouring defectors, its payoff may be smaller than 1, and on some certain graph structures will be too small a payoff for defection to spread. Such a graph structure includes the scenario where a cooperative neighbour has

much fewer non-cooperative neighbours than cooperative neighbours; this can provide the cooperators with a chance to prevent the invasion of defectors.

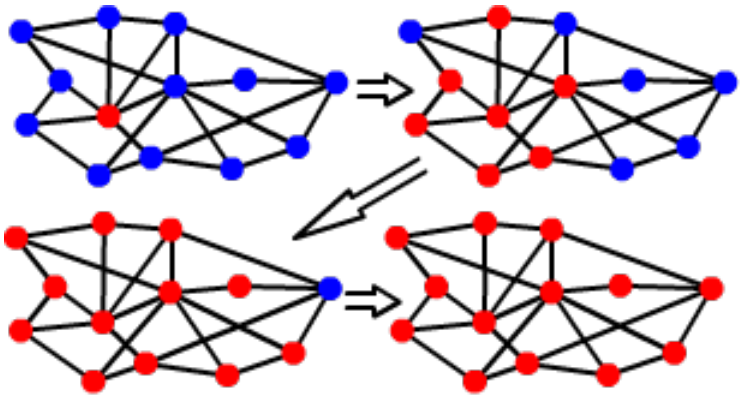

Fig. 5.2 The defectors invade the entire graph in a high degree graph

If the defector has n neighbours in total and m of them are also defectors, this defector's average payoff will be $(n-m)\beta + m\alpha$. For a neighbour of this defector, who has p neighbours but has only q defecting neighbours (including the defector it self), this neighbour will have the average payoff of $p-q$.

Therefore, a defector can only invade a cooperative neighbour when $(n-m)\beta + m\alpha > p-q$.

It is highly possible that $p-q > (n-m)\beta + m\alpha$, especially when those two agents do not have too many common neighbours. However, if one player in a fully cooperative complete sub graph (the size of the sub graphs are n) mutates to become a defector, all of its direct neighbours will learn to defect by the next generation.

However, on the bridge between two subgraphs, a defector will have $m = n-1$ defecting neighbours, so it will receive only $\beta + (n-1)\alpha$ as a payoff, which is much smaller than the payoff of its cooperative neighbour, who obtains a $n-1$ payoff due to its $n-1$ cooperative neighbours in its own complete subgraph. So, in this case, the defector will never invade another sub graph (Figure 5.3).

However, if the mutation happens to an agent on the bridge, it may invade 2 sub graphs, but no more (Figure 5.4).

According to the analysis, the cooperation rate, following the introduction of one defector in the graph, will be less likely to be reduced as the clustering coefficient increases (while keeping both size and the number of edges constant). Therefore, we hypothesise that a higher clustering coefficient is beneficial to the robustness of cooperation, as the higher clustering coefficient leads to a more clustered graph.

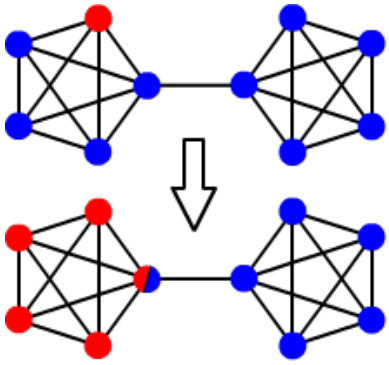

Fig. 5.3 One defector in the sub graph; as the vertex on the bridge linking to the other sub graph will get a smaller payoff than the cooperator in the other sub graph that is connected to it; it will continue swapping its strategy between cooperate and defect, and never invade other sub graphs.

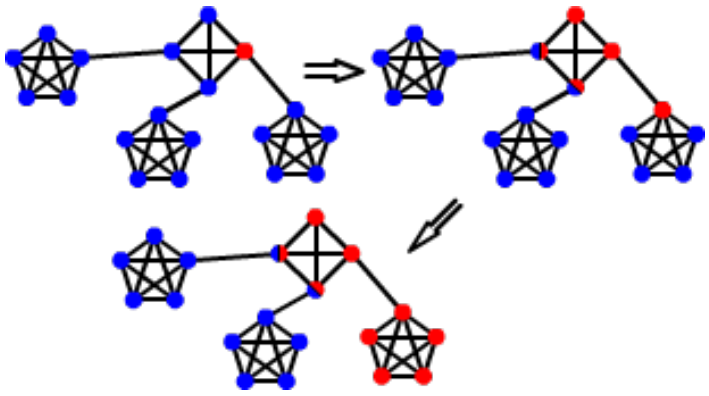

Fig. 5.4 The defector on the bridge may invade two sub graphs.

However, the number of edges in the graph also influences the structure of the graph, so in order to fully explore the relationship between the clustering coefficient and the robustness of cooperation in the experiments, a fixed number of edges is needed.

5.4 Experimental result

The game we used in these experiments is the prisoner's dilemma, with the following payoff matrix: $\begin{bmatrix} 1 & 0 \\ \beta & 0 \end{bmatrix}, \beta = 1.51$. This matrix has been used in much previous research.

In each generation, players will play a 2 player game against all of its direct neighbours and then receive the average payoff (as each player may have a different degree). They then learn the strategy that their most successful neighbour adopted. We adopt a simple learning model with no randomness so as to more easily understand the role each graph property plays in the evolution.

There are two main sets of experiments performed. First, we generate graphs with different clustering coefficients without controlling the number of edges, using the ascending graph generator. Second, we generate graphs with different clustering coefficients while guaranteeing a fixed number of edges.

The experiments will explore the influence of the clustering coefficient on the robustness of cooperation, and will also show whether the average degree influences the effect of the clustering coefficient.

5.4.1 Cooperation on graphs with different clustering coefficients without considering the average degree

The experiments are first undertaken on a set of graphs with 5000 vertices with the clustering coefficients varied from 0.1 to 0.6, generated by the ascending graph generation algorithm. The graph was initialized with all cooperators except for one random defector. We record the first 100 generations. Each experiment is the average value of 500 independent runs. The results are presented in Figure 5.5.

Figure 5.5 shows that when the clustering coefficient is equal to 0.1, 0.2, and 0.3, the cooperation rate is around 0.8, and the cooperation rate increases to 0.9 when the clustering coefficient is equal to 0.4, 0.5, and 0.6. Despite some fluctuations, a general trend can be observed.

It may be because the ascending graph generation algorithm adds more edges when

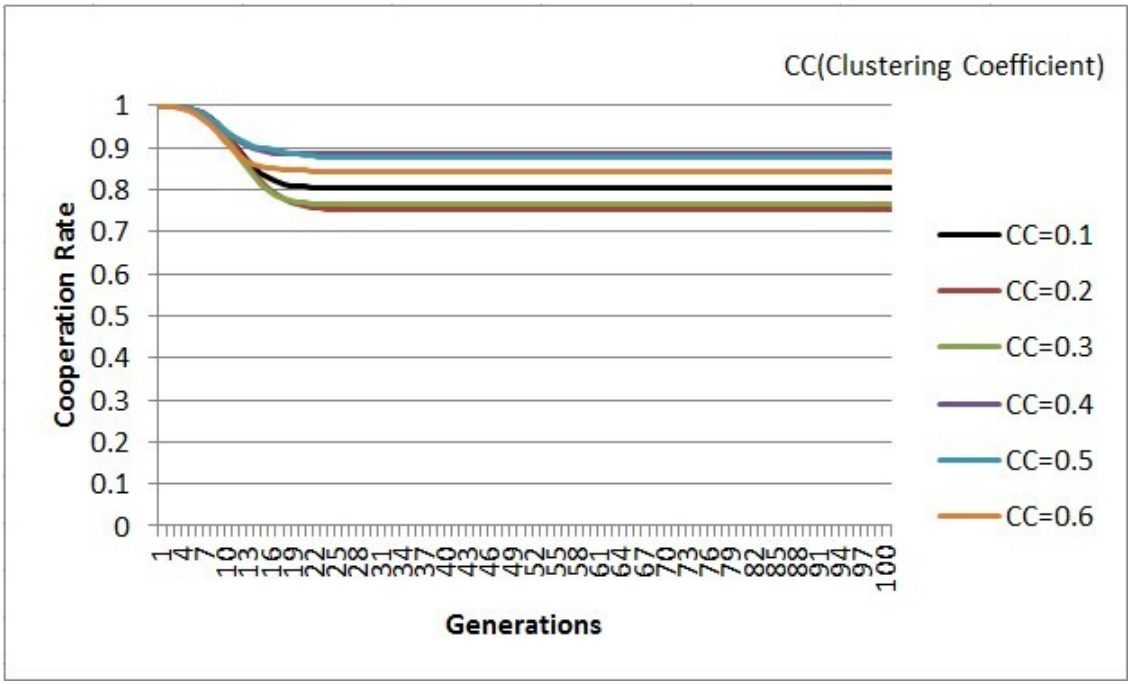

Fig. 5.5 Without controlling the number of edges, the experiment result is not significant enough to show the influence of the clustering coefficient on the robustness of cooperation.

attempting to increase the clustering coefficient. With the addition of more edges, the clustering coefficients could increase without changing the inherent structure of the graph in to a highly clustered graph which can prevent the spread of the defection.

However, to explore the effect of the clustering coefficient on the emergence of cooperation, we not only need to use the same size graph in terms of the number vertices but same size graphs in terms of both the number of vertices and the number of edges.

5.4.2 Cooperation on graphs with different clustering coefficients taking into account the average degree

The experiments above showed that the cooperation rate will decrease quickly as the average degree of the graph increases. To fully understand the relationship of the clustering coefficient and the robustness of cooperation, another experiment has been undertaken which uses a set of graphs generated by the heuristic graph generate algorithm, with 1000 vertices, and a relative smaller number of edges, 2500, and different clustering coefficients. According to the calculation in the last section, the graph with the average degree of 5, will has the maximum clustering coefficient around 0.86 (as the graph which has the architecture that is combined by a series of complete sub graphs with the size of 5, has the average degree 4.4,

has the maximum clustering coefficient $\frac{5^2-2\times5}{5^2-2\times5+2} \approx 8.6$). To obtain a sufficient sample, 10 different graphs have been generated for each clustering coefficient, from 0.1 to 0.8. For each graph, the experiments has been ran 100 times independently. So, for each clustering coefficient, the result is the average value of the cooperation level over 1000 independent runs. The result of the experiments is shown in the Figure 5.6

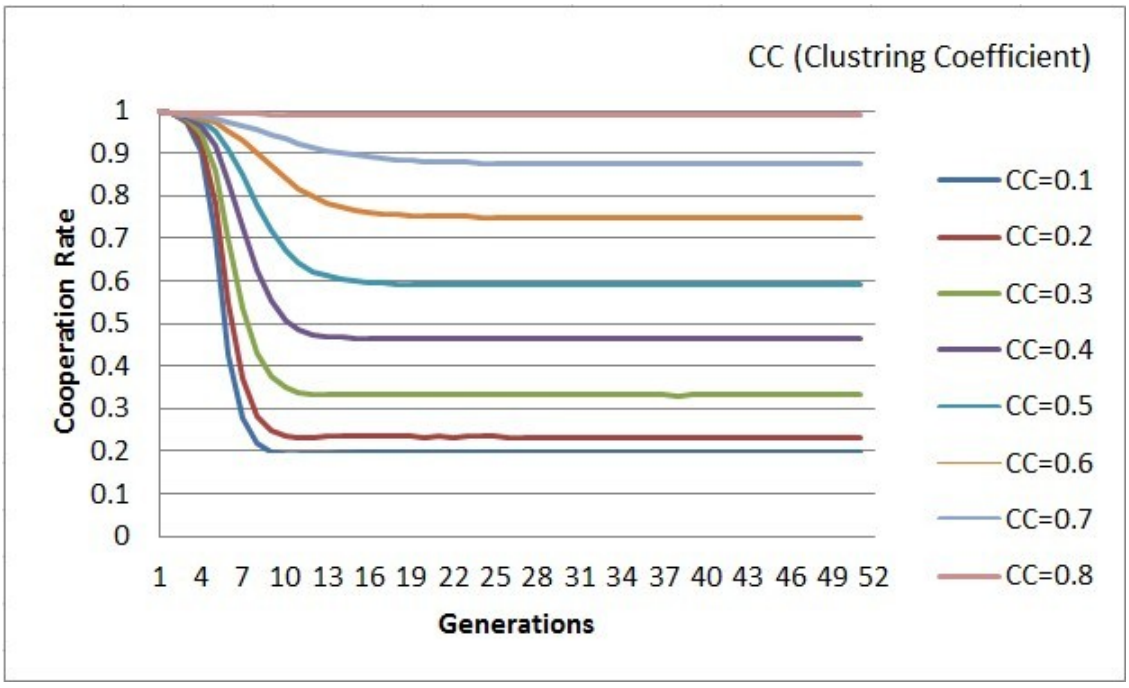

Fig. 5.6 With the same size, the cooperation rate of the graph is linearly increasing as the clustering coefficient increasing.

Figure 5.6 shows that with the same number of vertices and edges, the graph with high clustering coefficient, can benefit cooperation and prevent the spread of defectors.

5.5 Summary

This chapter presents two algorithms for generating graphs with a pre-defined clustering coefficient; the first constrains the number of numbers but not the number of edges; the second constrains both the number of nodes and the number of edges.

Simulations are used to explore how the average degree in the graph and the clustering coefficient of a graph influences the robustness of cooperation in a spatially organised population of agents.

The graph with a restricted number of edges and a high clustering coefficient is more

likely to be constructed as a series of highly clustering subgraphs, which could, in theory, benefit cooperation. Moreover, the maximum clustering coefficient of a graph with a certain number of vertices and edges can be estimated. We can also calculate the maximum number of edges that can be added to the graph while maintaining a clustering coefficient as 0 in a graph of a specific size.

As we expected, experiments show that in a connected graph of cooperators, if the average degree is high, the cooperators are not robust if one player mutates to be a defector. Conversely, with a limited average degree, graphs with a high clustering coefficient can provide better cooperation.

Chapter 6

Graph Centralities and the Invasion of the Defection

6.1 From Community to Individual

In the previous chapter, we have shown the graph topology plays an important role in the emergence of cooperation in evolutionary games [106]. Many researchers have studied how specific graph attributes can influence the emergence of cooperation by comparing the evolutionary results over graphs with different attributes. It is worth mentioning that even in the same graph, an individual player's neighbourhood structure can still influence significantly the behaviour of the evolution. Different neighbourhoods will perform differently with regards to the invasion of defection. Due to the high payoff possible through defection, a defector usually spreads quickly in graphs with high degree. Certain structural features of an individual's neighbourhood may halt the spread of defection and increase the robustness of cooperation for the entire graph, which has been considered as an important feature in some experimental data [112]. Understanding the individual's ability to stop the spread of defection will help to find the "weak" points of a graph, and it may be very useful in terms of virus protection, or advertising etc.

It is commonly believed that the hubs and communities in the graph have an important influence on cooperation [58, 64, 65, 103, 109]. Most research suggests that heterogeneous populations and hubs better support cooperation and there are also experiments that show that hubs and heterogeneous populations can guarantee to be of benefit to cooperation [58, 95]. Generally hubs and communities only consider the degree of one or many individuals; there may also be many other centrality measures of the individual that may also influence the emergence of cooperation. To fully understand cooperation on graphs, other

centrality measures, for example, the local clustering coefficient, closeness, betweenness and eigenvector centrality, are also worthy of consideration.

In this research, we explore the influence of different centrality measures on the behaviour of a population playing an evolutionary prisoner's dilemma, through identifying both "robust" and "weak" individuals in the graphs. After exploring a series of different graphs, we find that the graph transitivity can actually determine the number of highly robust individuals in low average degree graphs. We believe that individuals placed on nodes with a higher local clustering coefficient values but with lower degree values, lower betweenness scores and closeness individuals are more unlikely to spread defection.

Considering whether an individual is robust may also depend on its neighbour's centrality; in our experiments we do not fully explore this aspect; future work will explore this concept and with the aid of machine learning approaches, we hope to study and find the critical combination of the graph centralities that can predict the robustness of the individual (or "game centrality" [112]) in a graph.

In this chapter, we measure the spread of the defection from each individual over graphs with a wide range of average degree values and transitivity values; we observe several graph centrality measures and try to predict the spread of defection from an individual in the complex graph. In addition to considering the degree, clustering coefficient, closeness, betweenness, and degree eigenvector centrality, we identify potential candidates for correlation involving the combination of several measures.

6.2 Robustness of graph and robustness of individual

6.2.1 Cooperation of the graph and the invasion of individuals

In the last chapter, we have talked about how a community structure (or graph's attributes) can influence the cooperation, however, individual behaviour may also affect the cooperation of the entire graph. If a node has high ability to stop defection, that node is considered to be a "robust node", and a "robust graph" refers to a graph that was able to prevent the successful invasion of defectors. One measure of whether a graph is robust to the spread of cooperation is to examine the cooperation rate during an evolutionary run. It is true that different initial populations have an influence on the final result as a defector in different positions of the graph will influence the final cooperation rate of the evolution. The cooperation rate can be examined for a graph with different randomly initialized populations and the average calculated over several runs. Alternatively, one can use a set of pre-designed

patterns as initial populations.

Considering running an evolutionary simulation where players interact via a 2 player game with payoff matrix: $\begin{pmatrix} 1 & 0 \\ \beta & \alpha \end{pmatrix}$ in a complex graph. A defector who wishes to invade a group of cooperators, even in a strict learning environment (where the player will adopt the strategy of the most successful neighbour for the next generation), will need to gain a payoff score of more than 1 (which is the reward payoff obtained among mutual cooperators). As we know, in these 2 player games, the equilibrium is to defect. Hence, between each two individuals in the graph, the defector can always win the game against its direct cooperative neighbour. However, on a graph, both defector and cooperator will also be playing with other players. Since the defector is more likely to invade its cooperative neighbours, it may obtain a smaller *overall* payoff than cooperators who receive more rewards for mutual cooperation from interacting with fellow cooperators. Furthermore, for a mutator (i.e. a cooperator changed to a defector) in a fully cooperative population, the mutator will definitely obtain a better payoff due to all of its adjacent neighbours being cooperators (which means it will receive the temptation payoff in every game it plays). So, in a higher-degree graph, a single defector usually can invade most individuals due to the high connectivity in the graph. For this reason, in our research, the experiments are undertaken on graphs with a low average degree; this increases the likelihood that there will exist cooperative communities which can successfully avoid the invasion of defectors.

Previous research [62] showed that graph transitivity is a key to whether or not the cooperation level of the graph is high. However, this result is obtained using simulations over multiple runs over graphs with different transitivity values. In these simulations, one randomly chosen cooperator is changed to a defector and the effect is observed. Despite the general trend showing that graph transitivity is a key feature, there still exists exceptions where even in a very robust graph, defection may still invade a huge proportion of the graph. We posit that this is because those unusual cases involve nodes that exhibit some particular feature, which causes the outcome to differ from the majority outcome. In an effort to understand these features and phenomena, research on the individual node's robustness (or lack thereof) is necessary.

6.2.2 Invasion of a defector

Before we start discussing the spread of a defector in a graph, we need to know how a single defector can invade its neighbourhood. In our research, we adopt the mechanism whereby each player learns from her best performing neighbour, which means the

player will always learn from the neighbour with the highest payoff. Given a payoff matrix $\begin{pmatrix} 1 & 0 \\ \beta & \alpha \end{pmatrix}$, a defector will never invade a cooperator while it satisfies the following condition: $\frac{n\times\beta+(N-n)\times\alpha}{N} < \frac{m}{M}$, where N and M are the total number of neighbours of defector and cooperator respectively, n and m are the number of cooperative neighbours of the defector and cooperator respectively. To satisfy this condition, n must be much smaller than N.

In a fully cooperative graph, mutating one player from cooperation to defection in the initial generation, is equivalent to mutating all of its neighbours at the same time, as all of its neighbours will have learned to defect after just one generation. A high-degree graph is usually very poor at preventing the invasion of defection. This is the reason we restrict our experiments to low average-degree graphs.

This feature may in turn also stop the spread of defection in a highly clustered neighbourhood as a high local clustering coefficient of the first mutated node will spread defection causing a community of defectors to exist, which means they will receive a much lower payoff in the next generation as these connected defectors will receive the punishment payoff. So, we hypothesise that individual nodes with a higher clustering coefficient but a lower degree are more likely to stop the spread of defection.

6.2.3 Centrality measures that may influence the spread of defection

Considering the above analysis, in our experiments, in addition to the common centrality of the graph, the degree, the clustering coefficient, the betweenness, the closeness and the degree eigenvector centrality, we will also consider some combinations of these measures to see if they directly influence the ability of an individual to spread defection.

As we proposed earlier, a higher clustering coefficient but lower degree individual may be more likely to stop the spread of defection. So, we also consider $\frac{local\_clustering\_coefficient}{local\_degree}$ in the experiments. Furthermore, as the ability of an individual to spread defection may also be influenced by the individual's second and even third degree connections, we consider a measure for the clustering coefficient that takes this phenomenon into account: we consider the eigenvector of the degree. To calculate this, we first build a node-by-node matrix, M, where each cell $M_{i,j}$ has the value $\frac{common\_neighbours\_of\_node\_i\_and\_j}{total\_neighbours\_of\_node\_i\_and\_j}$. The eigenvector of this matrix can represent the centrality of the clustering of the corresponding individuals in the graph. We call this centrality measure the *clustering eigenvector centrality*.

As we consider $\frac{local\_clustering\_coefficient}{local\_degree}$, we similarly considered the value of $\frac{degree\_eigenvector}{clustering\_eigenvector}$ as a potential measure.

Since the spread of defection is very complex and may be controlled by more than two node centrality measures, we do not expect any of the measures to fully provide a perfect indication as to whether defection will spread or not. However, it may be give us a hint of how the graph centrality can influence an individual node's robustness with respect to cooperation. We hope this could provide a step towards developing a suitable prediction algorithm.

6.3 Experiment result

The experiments have been undertaken on a series of graphs with 1000 vertices with an average degree of 5. The graphs that have been generated for the experiment can be guaranteed to provide an even distribution of transitivity values. The experiments start with a fully cooperative population, and the player strategy updates are synchronized. At the beginning of each experiment, one individual is mutated to be a defector, and we then record the cooperation rate after 100 generations. For each graph we run the experiment 1000 times so that every individual is mutated once for an independent run in order to find which node will allow defection spread more widely.

We use 10 different graphs for each respective transitivity value from 0.1 to 0.8 which have been generated by the algorithm introduced in the paper [62]. We have also tested the graph with higher average degree values (6, 7, 8, 9 and 10) with each transitivity value from 0.1 to 0.8 as well. We decide to use a 2 player prisoner's dilemma game with a payoff matrix $\begin{pmatrix} 1 & 0 \\ 1.61 & 0 \end{pmatrix}$. The temptation payoff is set to 1.61, in order to prevent the case where a cooperator has the same payoff as a defector (for example, a defector with 5 cooperate neighbours will have payoff 8.05, that is slightly higher than a cooperator which has 8 cooperate neighbours).

We run each evolutionary simulation on each graph. The evolution converged very quickly and the cooperation rate reaches a relatively stable state usually within 50 generations. We recorded the cooperation rate of each independent run after 100 generations and used this as a measure of the robustness of cooperation for the corresponding individual.

We ranked the cooperation rate after 100 generations for each individual as a defector in each graph, and then we plotted several centrality measures of the individual, and attempted to identify if one (or a combination of a few) graph centrality measure can capture whether the individual will spread defection widely into the graph. We also explore for different graphs whether the graph's attributes can affect the overall performance of the individual in

the graph.

A sample plot of the cooperation rate for a graph with transitivity 0.5 (Graph 5.1) is shown below (Fig. 6.1).

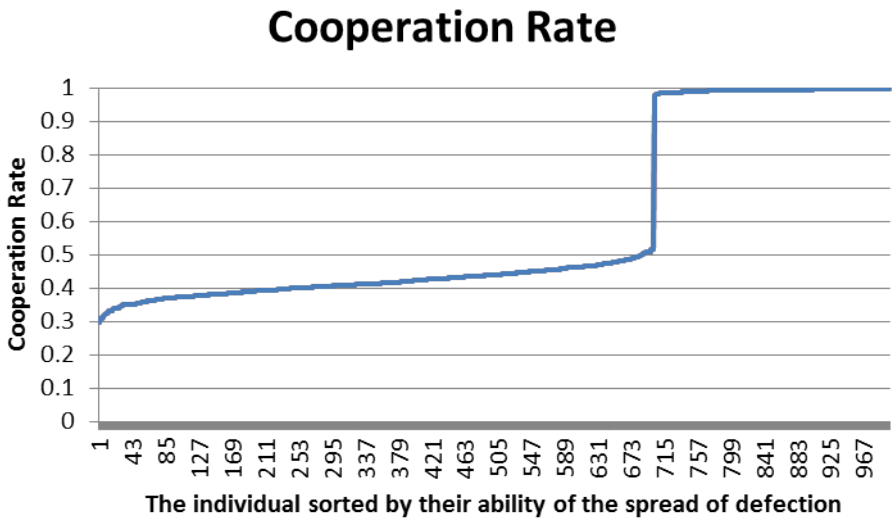

Fig. 6.1 The cooperation rate of each individual in graph with transitivity value 0.5

It is interesting to see that there are both "robust" and "weak" individuals in the same graph, with a quite clear separation (we can see that there is a quick transition from 0.52 to 0.98). We can see there is a transition that identifies the number of "robust" individuals and "weak" individuals with 705 individuals that will spread defection to more than half of the graph while others almost didn't spread defection at all.

This proportion varies in relation to the overall transitivity of the graph. The average rate of highly robust individuals averaged over 10 randomly generated graphs with transitivity values ranging from 0.1 to 0.6 (Fig. 6.2) shows that, for graphs which has 1000 vertices and 2500 edges, the higher the transitivity of the graph, the more robust individuals present in the graph. However, in the graphs with transitivity 0.7 and 0.8, there are hardly any individuals that allowed defection to spread very quickly; in other words, almost all individuals in such graphs are very robust. The overall comparison of graphs with different transitivity scores is shown in Figure 6.3.

From Figures 6.2 and 6.3, we can see that both the transitivity and the number of edges can determine the number of highly robust individuals in the graph. This supports previous results [62]. The individuals have quite different performance in each graph. Some of them spread defection over almost the entire graph, while others did not spread defection

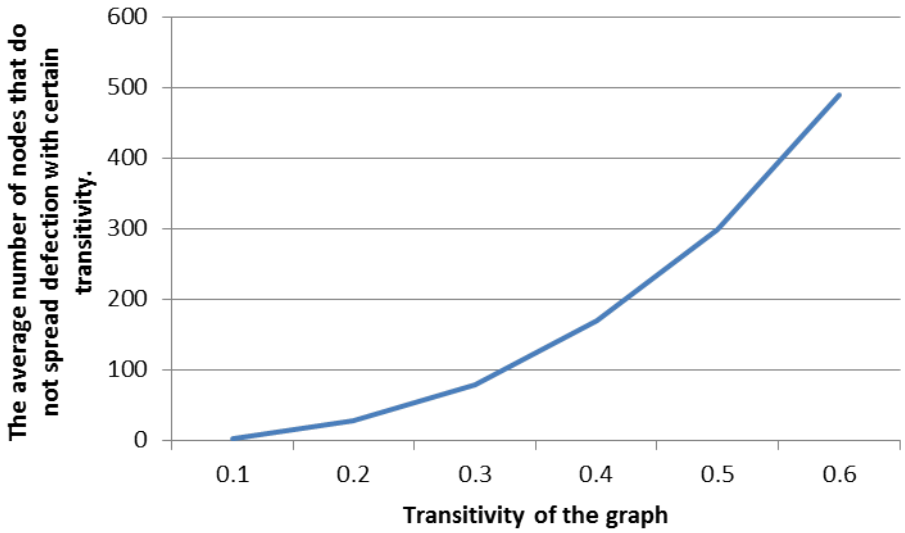

Fig. 6.2 The average number of individuals that can not spread defection in the graph with specific values of transitivity.

at all. In the graphs with high levels of transitivity, some individuals even reverted back to cooperation subsequent to their mutation to defection. To understand this phenomenon, we need to explore the centrality of each individual.

In order to observe whether there are any graph centrality measures that can identify the robustness of cooperation (or the spread of defectors) for each individual in the graph, we consider the following measures: the local clustering coefficient, local degree, betweenness centrality, closeness centrality and eigenvector centrality.

We plot the cooperation rate for each individual in each graph based on the rank for each different graph centrality measures. We include some sample plots in Figure 6.4 Figure 6.5 and Figure 6.6. These correspond to the graph illustrated in Figure 6.1. Other graphs demonstrate similar features.

From the experiments, we can see that there is a clear distinction for each centrality measure between the individuals leading to a low cooperation rate and those leading to a high cooperation rate. Although the results are quite noisy, there is a clear trend that a high clustering coefficient and high degree nodes are more robust to the spread of defection. Also, the betweenness and closeness scores are, on average, much lower for those robust nodes. Surprisingly, the influence of all of the eigenvector measures is quite small with little correlation between these values and the outcomes. This is probably due to the average degree of the graph being too small in comparison to the size of the graph, so the eigenvector's abso-

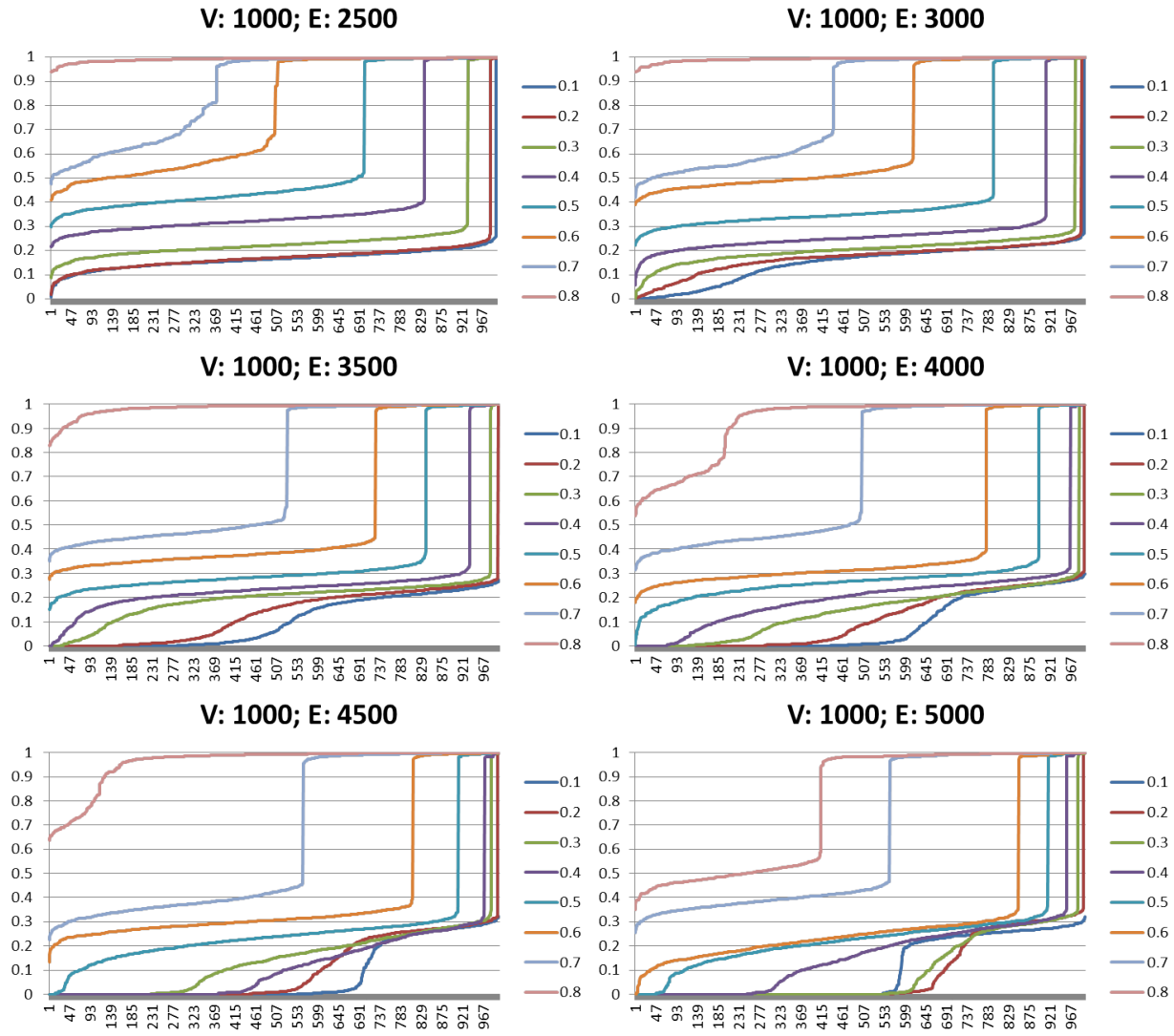

Fig. 6.3 Cooperation rates after 100 generations for the graphs with different transitivity and average degree. The number of edges is specified above the graphs; the colour of the lines defines the transitivity; the x-axis plots the index of the node that have been set to defect; the cooperation rate is plotted on the y-axis.

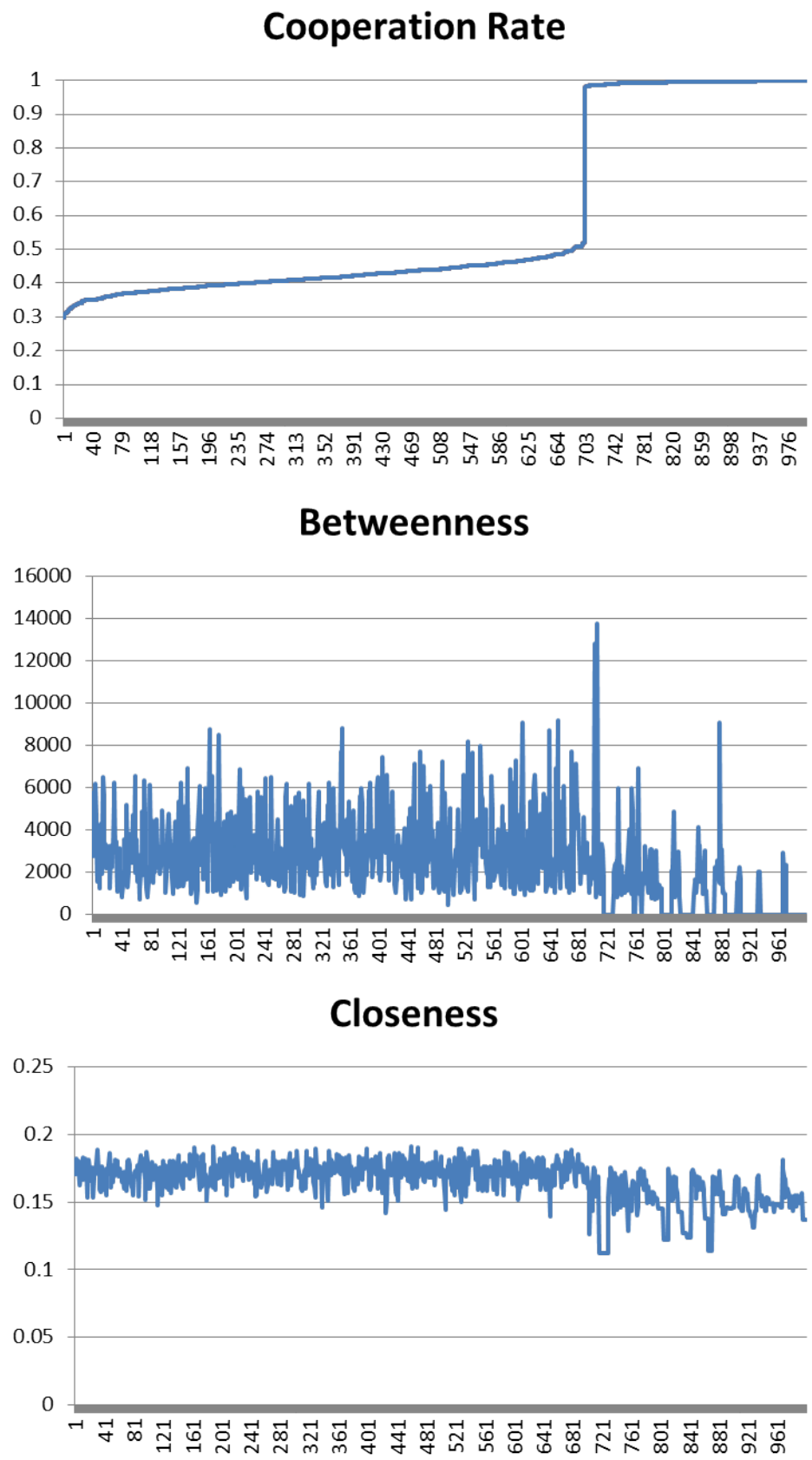

Fig. 6.4 Individual centrality by the order of each individual's spread of defection in graph 5.1, part 1

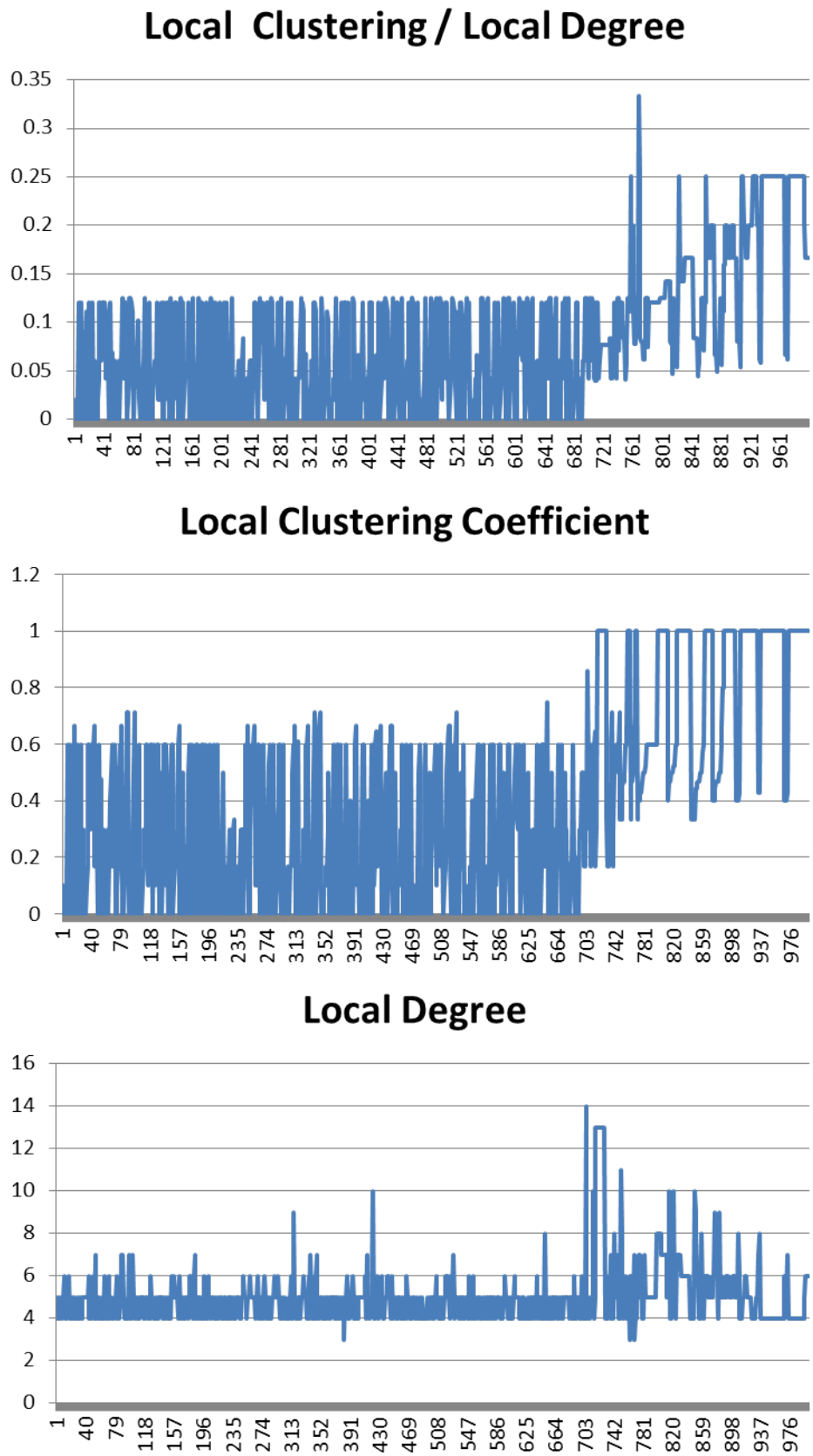

Fig. 6.5 Individual centrality by the order of each individual's spread of defection in graph 5.1, part 2

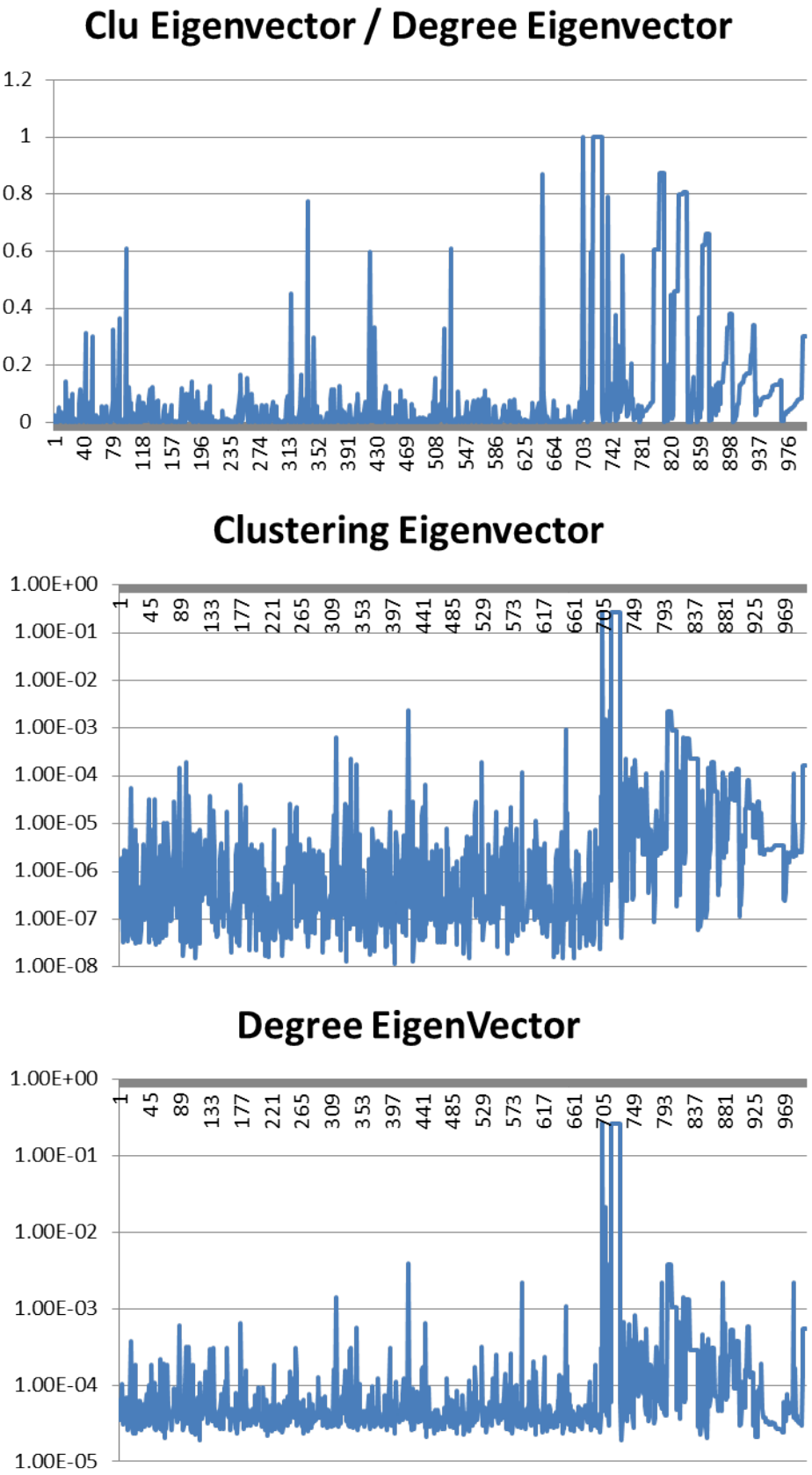

Fig. 6.6 Individual centrality by the order of each individual's spread of defection in graph 5.1, part 3

lute value is too small. Having said this, there is some correlation present with highly robust individuals having a higher probability of having a higher value of clustering eigenvector / degree eigenvector.

It is interesting to note that for individuals having a similar ability to spread defection, the variance on each centrality measure is high. There is an intermediate value of the centrality that may perform both behaviours of spreading defection or not.

However, the extreme values of some centrality measures, will still make a good prediction for the robustness of individuals. For example, a node with clustering coefficient 1, and betweenness 0, will definitely not spread defection at all.

The current findings are not strong enough to make a definite prediction on the individual robustness present in a graph as it is obvious that none of the graph centrality measures, on its own, exactly predict the spread of defection of the corresponding individual perfectly. However, the result gives us an indication of the effects of graph centrality on the spread of defection, as the extreme value only appears for either high or low cooperation.

6.4 Summary

By analysing each individual on a wide range of graphs, we attempt to make a connection between the graph topology and the cooperation levels present in social dilemma games from the "macro" level (the graph level) to the "micro" level (the individual view). The higher transitivity levels can decrease the number of nodes with a higher game centrality [112] (the nodes that have higher ability to spread defection), which extends the research in [62]. Increasing the average degree can increase the average game centrality since it actually increases the size of the community while retaining the actual graph size. Ignoring the size of the graph, we believe that cooperation on the graph can be represented as the spread of defection on each individual. The degree to which defection spreads from the individual decides the number of individuals that a defector can invade, which will directly influence the cooperation rate for graphs that have a finite number of vertices.

On the other hand, for an individual, the likelihood of defection spreading from the node can be roughly predicted by its centrality. Despite the presence of the high level of fluctuation in the intermediate values, the extreme values of the node centrality measure always appear with extreme values of the cooperation rate. So we can guarantee that a node with clustering coefficient 1 and betweenness 0, will never spread defection, even if that node itself changes to defection either through initial mutation or invasion.

Although the experiment results show some noise, the results are quite promising. Con-

sidering that the game centrality is a useful measure in many real world data settings, such as social networks, protein networs etc. [112], it cannot currently be predicted on given graphs without running the evolutionary social dilemma game on that particular graph. The measurable graph centrality is more easily obtained from the graph. Future work will involve learning the relationship between potential combinations of these centrality measures and the spread of defection. This hopefully will lead to a mechanism for predicting the spread of defection.

Chapter 7

Conclusion and Further Work

According to previous research, the factors that may influence the level of cooperation in an evolutionary game include: the game's payoff, the strategy set, the learning mechanism, the population structure and the graph topology and centralities. Understanding the influence of these factors can help us to understand more fully the evolutionary cooperation. Moreover, it may provide an approach to make predictions on the level of cooperation if we know those factors beforehand. To explore how each factor can affect the cooperation and to make prediction based on these factors, we have examined the level of cooperation in evolutionary games with respect to different aspects including the game's strategy and payoff and the graph's topology in each chapter respectively.

7.1 Conclusion

Through both mathematical analysis and simulation experiments, we found:

Given a payoff matrix and a specific initial population, the level of cooperation is predictable for games on a lattice

For evolutionary games where players are playing two-player pure-strategy games with Moore neighbourhood on lattice grids, we have built a prediction function based on each strategy's payoff value. Starting from the same initial population, if the learning is strict (there is no randomisation during the learning), the evolution will always the same. These will result in the same patterns in the visualisation. Also, in a lattice grid, those patterns that emerged are inconsistent with the value of the payoff; in other words, there are thresholds exist between each pair of adjacent patterns, fixed patterns will emergence for all payoff

values between two threshold values of the game payoff values. Due to the regularity of the lattice grid, the thresholds are only related to the average degree and the payoff values. Furthermore, the average degree only have influence on the number of the existing thresholds; the higher the average degree, the more patterns may exist in the evolution. So the values of the thresholds are only related to the values of the payoff matrix. Finding those thresholds is the key to making predictions of the level of cooperation using the payoff values. In this thesis, we have introduced an approach to find those thresholds by using a prediction function. Although the prediction function may create some redundant values (values that are not threshold values) in certain circumstances, however, it can guarantee to find all existing threshold values. With the prediction function, one can roughly predict the emergence of cooperation in lattices. This gives us an foresight of how can game's payoff influence the emergence of cooperation, which helped us to select appropriate payoff matrix in the rest of experiments in this thesis.

N-player TFT strategies can not guarantee to win an evolutionary game unless it have lower tolerance

The Tit For Tat (TFT) strategy has been shown to be one of the most successful strategies against defectors in two-player games. We have designed a series of N-player TFT strategies (TFTn) during this research. An attribute noted as "level of tolerance" has been introduced to identify different behaviours of the TFTn. Experiments have been designed to compare between pure C players, pure D players, and TFTn players with difference level of tolerance (TFT1 to TFT8). Result shows that in N-player games, TFT strategies can not guarantee to outperform a pure D strategy. However, we do find that the N-player TFT strategies with lower level of tolerance usually performs much better than others.

Cooperation are more robust on graphs that have lower degree and a higher transitivity

Regardless of the influence of the game's setting, it has been shown that the evolutionary game run on certain graphs performs better in terms of cooperation than those played on other graphs. These graphs were generally considered to have the feature of scale-free properties. However, there are no proofs regarding which attributes of the scale-free network induce highly robust cooperation. Rather than simply ascribe this feature to the power-law degree distribution, we have analysed both the degree and transitivity (total clustering coefficient) of the graph, and examined the related sub-graph communities. By running

simulations on graphs generated by our new graph generation algorithms (which are able to generate graphs with pre-defined degree and transitivity), we found that graphs with lower degree but higher transitivity are more robust to cooperation; in other words, one defector normally can only invade the community it belongs to, if the graph is constructed by many sub-communities that are close to a n-clique (the smaller n, the better), the graph will be more robust with respect to cooperation. Furthermore, we found that having same degree, the level of cooperation appears to be linearly related to the transitivity.

These findings showed that the reason for scale-free networks maintaining higher levels of cooperation. Moreover, we found that the defector cannot invade other nodes outside its own community if the community is similar to a n-clique (a community that closed to a complete subgraph with less amount of outgoing connections), which means not only in the scale-free networks, if any graph have such sub-communities, the defectors fall into that sub-community can not take over the cooperation of the entire graph. Follows the result of these experiments, a research on centralities of individual nodes in graph is necessary.

Graph centralities are important to the cooperation

Knowing that defectors on different nodes in the graphs may have different abilities to invade cooperators, we are trying to find out which centrality of the node can influence the invasion of the defection. We have examined several different centralities including the local degree, local clustering coefficient, closeness, betweenness, and eigenvector centralities. Experiments show that for a defector on a node that has a higher ability to invade others, that node usually has a higher probability to have a higher degree, clustering, and eigenvector centrality, while its betweenness and closeness are more likely to be lower. This result is unrelated to the topology the entire graph, which means, the ability of a defector to invade may be predictable by analysis of the defector's node's centralities. Overall, the experiment shows a quite promising result of the relation between individual centrality and the invasion of defectors.

The most significant contribution of this work is that we bring the research of the graph topology in evolutionary games into the "micro" level (the individual view), this will help us to understand how exactly defectors can invade cooperators, and the reasons why the structure of the graph can maintain the cooperation, without analysing the entire graph's topology. The result may help us to make predictions on the level of cooperation by calculating only the defector's centralities, and it works on both finite and infinite graphs.

Summary

This thesis has discussed the cooperative outcomes of the evolutionary games on complex graphs. We have concluded the main factors that can influence the level of cooperation include:

1. The attribute of the game (payoffs, two or multi player etc.).
2. Player's strategy.
3. The learning between players.
4. Population structure.
5. Individual node's centrality.

We have analysed the influence of the payoff of the game and iterated strategies, then selected a setting of those factors that have minimum influence while research the effects of graph topology on the cooperation of evolutionary games. During the research, we found that the ability of defectors to invade be based on the defector's node's centralities.

7.2 Future Work

This thesis discussed the influence of different factors, especially the graph topology on evolutionary games. During the research, we have collected lots of graph centrality data from individual nodes that have different abilities to defect. Although we believe that prediction of the spread of defection can be made from a single defector's graph centralities. However, this prediction formula is hard to find mathematically. Since the prediction can not be made through a single centrality, more rigorous statistical analysis is needed to find this relation. Apart from this, the research on the individual node's influence on cooperation has gave us a prospective of the research on infinite graphs since we only need to analysis the sub-communities which include defectors. Also, the research in this work shows the protencial to calculate the level of cooperation using mathematics formula, the evolutionary games can be used to simulate many real-world problems, such as the spread of virus, or advertising on social network etc., this work could be extend to many real-world area, and the experiments in this research shows quite promissing result on the potencial to solve those problems.

Besides the real-world problem solving, the theoritical background can also be extended. In this research, we have shows how different property can influence the cooperation, for

example, the influence of the graph attributes and centralities in cooperation, and the iterated strategies in lattice grid. However, furthure work might considering to research the influence of the combination of those attributes. Also, more game and graph parameters's influence might worth to be developed.

References

[1] Allen, B. and Nowak, M. A. (2013). Cooperation and the fate of microbial societies. *PLoS Biol*, 11(4):e1001549.

[2] Ashlock, D. (2007). Cooperation in prisoner's dilemma on graphs. In *Computational Intelligence and Games, 2007. CIG 2007. IEEE Symposium on*, pages 48–55.

[3] Ashlock, D., McGuinness, C., and Ashlock, W. (2012). Representation in evolutionary computation. In Liu, J., Alippi, C., Bouchon-Meunier, B., Greenwood, G., and Abbass, H., editors, *Advances in Computational Intelligence*, volume 7311 of *Lecture Notes in Computer Science*, pages 77–97. Springer Berlin Heidelberg.

[4] Assenza, S., Gómez-Gardeñes, J., and Latora, V. (2008). Enhancement of cooperation in highly clustered scale-free networks. *Phys. Rev. E*, 78:017101.

[5] Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. *Genetic algorithms and simulated annealing*, pages 32–41.

[6] Axelrod, R. and Hamilton, W. D. (1981). *The Evolution of Cooperation*.

[7] Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

[8] Barlow, L.-A. and Ashlock, D. (2013). The impact of connection topology and agent size on cooperation in the iterated prisoner's dilemma. In *CIG*, pages 1–8.

[9] Barry, J. M., Hatfield, J. W., and Kominerst, S. D. (2013). On derivatives markets and social welfare: A theory of empty voting and hidden ownership. *Virginia Law Review*, 99(6):1103 – 1167.

[10] Bassett, D. S. S. and Bullmore, E. (2006). Small-world brain networks. *The Neuroscientist : a review journal bringing neurobiology, neurology and psychiatry*, 12(6):512–523.

[11] Bavelas, A. (1948). A mathematical model for group structures. *Human Organization*, 7:16–30.

[12] Bavelas, A. (1950). Communication Patterns in Task-Oriented Groups. *Journal of The Acoustical Society of America*, 22.

[13] Biggs, N., Lloyd, E., and Wilson, R. (1736-1936). *Graph Theory*. Oxford University Press.

[14] Bonacich, P. (1972). Factoring and weighting approaches to status scores and clique identification. *The Journal of Mathematical Sociology*, 2(1):113–120.

[15] Bonacich, P. (1987). Power and centrality: A family of measures. *American Journal of Sociology*, 92(5):pp. 1170–1182.

[16] Bonacich, P. (2007). Some unique properties of eigenvector centrality. *Social Networks*, 29(4):555 – 564.

[17] Bonacich, P. and Lloyd, P. (2001). Eigenvector-like measures of centrality for asymmetric relations. *Social Networks*, 23(3):191 – 201.

[18] Boyd, R. and Richerson, P. (1988). The evolution of reciprocity in sizable groups. *Journal of Theoretical Biology*, 132(3):337–356.

[19] Boyd, R. and Richerson, P. J. (2002). Group beneficial norms can spread rapidly in a structured population. *Journal of theoretical biology*, 215(3):287–296.

[20] Bozic, I., Allen, B., and Nowak, M. A. (2012). Dynamics of targeted cancer therapy. *Trends in Molecular Medicine*, 18(6):311 – 316.

[21] Brandes, U. (2001). A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177.

[22] Chen, X., Fu, F., and Wang, L. (2007). Prisoner's Dilemma on community networks. *Physica A: Statistical Mechanics and its Applications*, 378(2):512–518.

[23] Chiong, R., Dhakal, S., and Jankovic, L. (2007). Effects of neighbourhood structure on evolution of cooperation in n-player iterated prisoner's dilemma. In *Proceedings of the 8th international conference on Intelligent data engineering and automated learning*, IDEAL'07, pages 950–959, Berlin, Heidelberg. Springer-Verlag.

[24] Chiong, R. and Kirley, M. (2011). Iterated n-player games on small-world networks. In *GECCO*, pages 1123–1130.

[25] Chiong, R. and Kirley, M. (2012a). Effects of iterated interactions in multiplayer spatial evolutionary games. *IEEE Trans. Evolutionary Computation*, 16(4):537–555.

[26] Chiong, R. and Kirley, M. (2012b). Random mobility and the evolution of cooperation in spatial -player iterated prisoner's dilemma games. *Physica A: Statistical Mechanics and its Applications*, 391(15):3915 – 3923.

[27] Du, W.-B., Cao, X.-B., and Hu, M.-B. (2009). The effect of asymmetric payoff mechanism on evolutionary networked prisoner's dilemma game. *Physica A: Statistical Mechanics and its Applications*, 388(24):5005–5012.

[28] Eisert, J. and Wilkens, M. (2000). Quantum games. *Journal of Modern Optics*, 47(14-15):2543–2556.

[29] Eisert, J., Wilkens, M., and Lewenstein, M. (1999). Quantum games and quantum strategies. *Physical Review Letters*, 83(15):3077.

[30] Erdos, P. and Renyi, A. (1959). On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297.

[31] Fogel, D. B. (1993). Evolving behaviors in the iterated prisoner's dilemma. *Evol. Comput.*, 1(1):77–97.

[32] Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. John Wiley, New York, USA.

[33] Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174.

[34] Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 40(1):pp. 35–41.

[35] Freeman, L. C. (1978). Centrality in social networks conceptual clarification. *Social Networks*, page 215.

[36] Fu, F., Liu, L. H., and Wang, L. (2007). Evolutionary Prisoner's Dilemma on heterogeneous Newman-Watts small-world network. *The European Physical Journal B - Condensed Matter and Complex Systems*, 56(4):367–372.

[37] Fu, F., Nowak, M. A., and Hauert, C. (2010). Invasion and expansion of cooperators in lattice populations: prisoner's dilemma vs. snowdrift games. *Journal of theoretical biology*, 266(3):358–366.

[38] Fu, F., Tarnita, C. E., Christakis, N. A., Wang, L., Rand, D. G., and Nowak, M. A. (2012). Evolution of in-group favoritism. *Scientific Reports*.

[39] Fukami, T. and Takahashi, N. (2014). New classes of clustering coefficient locally maximizing graphs. *Discrete Applied Mathematics*, 162(0):202 – 213.

[40] Gintis, H. (2000). *Game Theory Evolving*. Princeton University Press.

[41] Gómez, D., González-Arangüena, E., Manuel, C., Owen, G., del Pozo, M., and Tejada, J. (2003). Centrality and power in social networks: a game theoretic approach. *Mathematical Social Sciences*, 46(1):27 – 54.

[42] Grofman, B. and Owen, G. (1982). A game theoretic approach to measuring degree of centrality in social networks. *Social Networks*, 4(3):213 – 224.

[43] Guo, Q., Zhou, T., Liu, J.-G., Bai, W.-J., Wang, B.-H., and Zhao, M. (2006). Growing scale-free small-world networks with tunable assortative coefficient. *Physica A: Statistical Mechanics and its Applications*, 371(2):814 – 822.

[44] Hanaki, N., Peterhansl, A., Dodds, P. S., and Watts, D. J. (2007). Cooperation in evolving social networks. *Management Science*, 53(7):1036–1050.

[45] Hauert, C. (2006). Spatial effects in social dilemmas. *Journal of Theoretical Biology*, 240(4):627–636.

[46] Hauert, C. and Imhof, L. A. (2012). Evolutionary games in deme structured, finite populations. *Journal of Theoretical Biology*, 299(0):106 – 112. Evolution of Cooperation.

[47] Heath, L. S. and Parikh, N. (2011). Generating random graphs with tunable clustering coefficients. *Physica A: Statistical Mechanics and its Applications*, 390(23–24):4577 – 4587.

[48] Herrera, C. and Zufiria, P. J. (2011). Generating scale-free networks with adjustable clustering coefficient via random walks. In *Proceedings of the 2011 IEEE Network Science Workshop*, NSW '11, pages 167–172, Washington, DC, USA. IEEE Computer Society.

[49] Hill, A., Rosenbloom, D., and Nowak, M. (2012). Evolutionary dynamics of hiv at multiple spatial and temporal scales. *Journal of Molecular Medicine*, 90(5):543–561.

[50] Holland, J. H. (1992). Genetic algorithms. *Scientific American*, pages 66–72.

[51] Holme, P. and Ghoshal, G. (2006). Dynamics of Networking Agents Competing for High Centrality and Low Degree. *Physical Review Letters*, 96(9):098701.

[52] Holme, P. and Kim, B. J. (2002). Growing scale-free networks with tunable clustering. *Physical Review E*, 65(2):026107.

[53] Howley, E. and O'Riordan, C. (2008). The effects of payoff preferences on agent tolerance. In *ALIFE*, pages 249–256.

[54] Ichinose, G., Tenguishi, Y., and Tanizawa, T. (2013). Robustness of cooperation on scale-free networks under continuous topological change. *Physical Review E*, 88(5).

[55] John C, H., W, L., and Eckehart, K. (1998). *Game Theory, Experience, Eationality: Foundations of Socialism*. Springer.

[56] Langer, P., Nowak, M. A., and Hauert, C. (2008). Spatial invasion of cooperation. *Journal of Theoretical Biology*, 250(4):634–641.

[57] LEI Chuang, JIA Jian-Yuan, C. X.-J., Rui, C., and Long, W. (2009). Prisoner's dilemma game on clustered scale-free networks under different initial distributions. *Chinese Physics Letters*, 26(8):80202.

[58] Li, C., Zhang, B., Cressman, R., and Tao, Y. (2013a). Evolution of cooperation in a heterogeneous graph: Fixation probabilities under weak selection. *PLoS ONE*, 8(6):e66560.

[59] Li, J. and Kendall, G. (2009). A strategy with novel evolutionary features for the iterated prisoner's dilemma. *Evolutionary Computation*, 17(2):257–274.

[60] Li, J. and Kendall, G. (2010). Collective behavior and kin selection in evolutionary ipd. *Journal of Multiple-Valued Logic and Soft Computing*, 16(6):509–525.

[61] Li, J., Kendall, G., and Vasilakos, A. V. (2013b). Backward induction and repeated games with strategy constraints: An inspiration from the surprise exam paradox. *IEEE Transactions on Computational Intelligence and AI in Games*, 5(3):242–250.

[62] Li, M. and O'Riordan, C. (2013). The effect of clustering coefficient and node degree on the robustness of cooperation. In *IEEE Congress on Evolutionary Computation*, pages 2833–2839.

[63] Lieberman, E., Hauert, C., and Nowak, M. A. (2005). Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316.

[64] Luthi, L., Pestelacci, E., and Tomassini, M. (2008). Cooperation and community structure in social networks. *Physica A: Statistical Mechanics and its Applications*, 387(4):955 – 966.

[65] Maciejewski, W., Fu, F., and Hauert, C. (2014). Evolutionary game dynamics in populations with heterogeneous structures. *PLoS Computational Biology*, 10(4).

[66] Macy, M. W. and Flache, A. (2002). Learning dynamics in social dilemmas. *Proceedings of the National Academy of Sciences of the United States of America*, 99(10):pp. 7229–7236.

[67] Majeski, S. J., Linden, G., Linden, C., and Spitzer, A. (1999). Agent mobility and the evolution of cooperative communities. *Complexity*, 5(1):16–24.

[68] Maynard Smith, J. (1982). *Evolution and the theory of games*. Cambridge University Press, Cambridge ; New York.

[69] Meyer, D. (2001). Quantum games and quantum algorithms. *AMS Contemporary Mathematics*, 5.

[70] Meyer, D. A. (1998). Quantum strategies. *Physical Review Letters*, 82:1052–1055.

[71] Miyaji, K., Tanimoto, J., Wang, Z., Hagishima, A., and Ikegaya, N. (2013). Direct reciprocity in spatial populations enhances r-reciprocity as well as st-reciprocity. *PLoS ONE*, 8(8):e71961.

[72] Mokken, R. J. (1979). Cliques, clubs and clans. *Quality and Quantity*, 13(2):161–173.

[73] Molinero, X., Riquelme, F., and Serna, M. (2013). Power indices of influence games and new centrality measures for social networks.

[74] Molloy, M. and Reed, B. (1995). A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms*, 6(2-3):161–180.

[75] Nash, J. F. (1950). Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences of the United States of America*, 36(1):48–49.

[76] Nash, J. F. (1951). Non-cooperative games. *Annals of Mathematics*, 54(2):286–295.

[77] Newman, M. E. J. (2009). Random graphs with clustering. *Physical Review Letters*, 103:058701.

[78] Nowak, M. A. (2006). Five rules for the evolution of cooperation. *Science*, 314(5805):1560–1563.

[79] Nowak, M. A., Bonhoeffer, S., and May, R. M. (1994a). More spatial games. *Int J Bifurcat Chaos*, 4:33–56.

[80] Nowak, M. A., Bonhoeffer, S., and May, R. M. (1994b). Spatial games and the maintenance of cooperation. *Proceedings of the National Academy of Sciences of the United States of America*, 91(11):4877–4881.

[81] Nowak, M. A. and May, R. M. (1992). Evolutionary games and spatial chaos. *Nature*, 359:826.

[82] Nowak, M. A. and May, R. M. (1993). The spatial dilemmas of evolution. *International Journal of Bifurcation and Chaos (IJBC)*, 3(1):35–78.

[83] Nowak, M. a., Tarnita, C. E., and Wilson, E. O. (2010). The evolution of eusociality. *Nature*, 466(7310):1057–1062.

[84] Opsahl, T. and Panzarasa, P. (2009). Clustering in weighted networks. *Social Networks*, 31(2):155–163.

[85] O'Riordan, C., Cunningham, A., and Sorensen, H. (2008). Emergence of cooperation in n-player games on small world networks. In *ALIFE*, pages 436–442.

[86] O'Riordan, C., Curran, D., and Sorensen, H. (2007). Spatial n-player dilemmas in changing environments. In *SGAI Conf.*, pages 381–386.

[87] O'Riordan, C., Griffith, J., Curran, D., and Sorensen, H. (2006). Norms and cultural learning in the N-player prisoner's dilemma. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 1105 –1110.

[88] O'Riordan, C., Griffith, J., Newell, J., and Sorensen, H. (2004). Co-evolution of strategies for an N-player dilemma. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 2, pages 1625 – 1630 Vol.2.

[89] O'Riordan, C. and Sorensen, H. (2007). Stable cooperation in the n-player prisoner's dilemma: The importance of community structure. In *Adaptive Agents and Multi-Agents Systems*, pages 157–168.

[90] Pacheco, J. M., Pinheiro, F. L., and Santos, F. C. (2009). Population structure induces a symmetry breaking favoring the emergence of cooperation. *PLoS Computational Biology*, 5(12):1.

[91] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

[92] Perc, M. (2009). Evolution of cooperation on scale-free networks subject to error and attack. *New Journal of Physics*, 11:033027.

[93] Perc, M. and Szolnoki, A. (2010). Coevolutionary games a mini review. *Biosystems*, 99(2):109 – 125.

[94] Poncela, J., Gómez-Gardeñes, J., Floría, L. M., and Moreno, Y. (2007). Robustness of cooperation in the evolutionary prisoner's dilemma on complex networks. *New Journal of Physics*, 9(6):184.

[95] Poncela, J., Gómez-Gardeñes, J., Floría, L. M., Moreno, Y., and Sánchez, A. (2009). Cooperative scale-free networks despite the presence of defector hubs. *EPL (Europhysics Letters)*, 88(3):38003.

[96] Poncela, J., Gómez-Gardeñes, J., Moreno, Y., and Floría, L. M. (2010). Cooperation in the prisoner's dilemma game in random scale-free graphs. *International Journal of Bifurcation and Chaos*, 20(03):849.

[97] Price, D. D. S. (1976). A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27(5):292–306.

[98] Qin, S.-M., Zhang, G.-Y., and Chen, Y. (2009). Coevolution of game and network structure with adjustable linking. *Physica A*, 388(23):4893.

[99] Rand, D. G. and Nowak, M. A. (2013). Human cooperation. *Trends in Cognitive Sciences*, 17(8):413 – 425.

[100] Rapoport, A. (1953). Spread of information through a population with socio-structural bias: I. assumption of transitivity. *The bulletin of mathematical biophysics*, 15(4):523–533.

[101] Ren, J., Wang, W.-X., and Qi, F. (2007). Randomness enhances cooperation: A resonance-type phenomenon in evolutionary games. *Phys. Rev. E*, 75:045101.

[102] Rezaei, G. and Kirley, M. (2012). Dynamic social networks facilitate cooperation in the n-player prisonerr's dilemma. *Physica A: Statistical Mechanics and its Applications*.

[103] Roos, P., Gelfand, M., Nau, D., and Carr, R. (2014). High strength-of-ties and low mobility enable the evolution of third-party punishment. *Proceedings of the Royal Society B: Biological Sciences*, 281(1776).

[104] Rui, C., Yuan-Ying, Q., Xiao-Jie, C., and Long, W. (2010). Robustness of cooperation on highly clustered scale-free networks. *Chinese Physics Letters*, 27(3):30203.

[105] Sabidussi, G. (1966). The centrality index of a graph. *Psychometrika*, 31(4):581–603.

[106] Santos, F., Rodrigues, J., and Pacheco, J. (2006a). Graph topology plays a determinant role in the evolution of cooperation. *Proceedings of the Royal Society B: Biological Sciences*, 273(1582):51–55.

[107] Santos, F. C. and Pacheco, J. M. (2005). Scale-free networks provide a unifying framework for the emergence of cooperation. *Phys. Rev. Lett.*, 95:098104.

[108] Santos, F. C., Pacheco, J. M., and Lenaerts, T. (2006b). Evolutionary dynamics of social dilemmas in structured heterogeneous populations. *Proc. Natl. Acad. Sci. USA*, 103:3490–3494.

[109] Santos, F. C., Santos, M. D., and Pacheco, J. M. (2008). Social diversity promotes the emergence of cooperation in public goods games. *Nature*, 454(7201):213–216.

[110] Shigaki, K., Wang, Z., Tanimoto, J., and Fukuda, E. (2013). Effect of initial fraction of cooperators on cooperative behavior in evolutionary prisoner's dilemma game. *PLoS ONE*, 8(11):e76942.

[111] Sicardi, E. A., Fort, H., Vainstein, M. H., and Arenzon, J. J. (2008). Random mobility and spatial structure often enhance cooperation. *Journal of theoretical biology*, 240(6):256.

[112] Simko, G. I. and Csermely, P. (2013). Nodes having a major influence to break cooperation define a novel centrality measure: Game centrality. *PLoS ONE*, 8(6):e67159.

[113] Smith, J. M. and Price, G. R. (1973). The Logic of Animal Conflict. *Nature*, 246(5427):15–18.

[114] Spizzirri, L. (2011). Justification and application of eigenvector centrality. *Algebra in Geography: Eigenvectors of Networks*, 7.

[115] Stephenson, K. and Zelen, M. (1989). Rethinking centrality: Methods and examples. *Social Networks*, 11(1):1–37.

[116] Szolnoki, A., Perc, M., and Danku, Z. (2008a). Making new connections towards cooperation in the prisoner's dilemma game. *EPL (Europhysics Letters)*, 84(5):50007.

[117] Szolnoki, A., Perc, M., and Danku, Z. (2008b). Towards effective payoffs in the prisoner's dilemma game on scale-free networks. *Physica A: Statistical Mechanics and its Applications*, 387(8-9):2075–2082.

[118] Tang, C.-L., Wang, W.-X., Wu, X., and Wang, B.-H. (2006). Effects of average degree on cooperation in networked evolutionary game. *The European Physical Journal B - Condensed Matter and Complex Systems*, 53(3):411–415.

[119] Tanimoto, J. (2007). Dilemma solving by the coevolution of networks and strategy in a 2×2 game. *Phys. Rev. E*, 76:021126.

[120] Taylor, P. (1978). Evolutionary stable strategies and game dynamics. *Mathematical Biosciences*, 40(1-2):145–156.

[121] van Gestel, J., Nowak, M. A., and Tarnita, C. E. (2012). The evolution of cell-to-cell communication in a sporulating bacterium. *PLoS Comput Biol*, 8(12):e1002818.

[122] Van Segbroeck, S., de Jong, S., Nowé, A., Santos, F. C., and Lenaerts, T. (2010). Learning to coordinate in complex networks. *Adaptive Behavior*, 18(5):416–427.

[123] von Neumann, J. and Morgenstern, O. (2004). *Theory of Games and Economic Behavior (Commemorative Edition) (Princeton Classic Editions)*. Princeton University Press.

[124] Vukov, J., Szaboacute, G., and Szolnoki, A. (2008). Evolutionary prisoner's dilemma game on newman-watts networks. *Phys Rev E Stat Nonlin Soft Matter Phys*, 77(2 Pt 2):026109.

[125] Wang, Z., Szolnoki, A., and Perc, M. (2012). If players are sparse social dilemmas are too: Importance of percolation for evolution of cooperation. *Scientific Reports*, 2.

[126] Watts, D. J. (1999). *Small worlds: the dynamics of networks between order and randomness*. Princeton university press.

[127] Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of small-world networks. *Nature*, 393(6684):440–442.

[128] Weibull, J. (1995). *Evolutionary Game Theory*. MIT Press, Cambridge ; Massachusetts:.

[129] Wu, B., Zhou, D., Fu, F., Luo, Q., Wang, L., and Traulsen, A. (2010). Evolution of Cooperation on Stochastic Dynamical Networks. *PLoS ONE*, 5(6):e11187.

[130] Wu, Z., Xu, X., Chen, Y., and Wang, Y. (2005). Spatial prisonerr's dilemma game with volunteering in newman-watts small-world networks. *Physical Review E*, 71(3):037103.

[131] WU Gang, GAO Kun, Y. H.-X. and Bing-Hong, W. (2008). Role of clustering coefficient on cooperation dynamics in homogeneous networks. *Chinese Physics Letters*, 25(6):2307.

[132] Yang, D.-P., Lin, H., Wu, C.-X., and Shuai, J. (2011). Topological conditions of scale-free networks for cooperation to evolve.

[133] Yao, X. and Darwen, P. J. (1995). An experimental study of N-person iterated prisoner's dilemma games. In *Selected papers from the AI'93 and AI'94 Workshops on Evolutionary Computation, Process in Evolutionary Computation*, AI '93/AI '94, pages 90–108, London, UK, UK. Springer-Verlag.

Appendix A

The Graph Simulator : A User Manual

A.1 Introduction

Appendix 1 is the user manual of the graph simulator that I have created for the experiments in this thesis. Section 2 listed the system requirements to run the simulator; Section 3 demonstrated how to use the simulator; Section 4 listed the hot keys of the simulator.

A.2 System Requirements

A.2.1 Running the Simulator

Windows 7 or higher system

The current version of the graph simulator is developed using DirectX 11. On windows 7, you need to install Microsoft DirectX 11 runtime to run the simulator. DirectX 11 is installed by default on Windows 8 or higher.

Windows XP or lower system

As Direct 11 is not supported on any system lower than Windows 7, you can not install directX 11 on windows XP. An older version using DirectX 9 is provided as an alternative. To run that on Windows XP, you need to install the DirectX 9 runtime instead. This version is no longer maintened and it may not include all features.

Other systems

As the graph simulator is developed using a C++ / DirectX environment, it only works on Microsoft Windows systems. If you want to use it on other system, please download the source code, and rewrite the graphic framework part using OpenGL.

A.2.2 Compiling the Source Code

This is an open source project, so you can download all the source code from our website.

Compiling with Visual Studio 2012 or higher

1. Install Microsoft Visual Studio Express: you can download the latest version of Visual Studio Express from Microsoft's website for free.

2. Install DirectX SDK: if you're compiling the DX11 version, you need to download the latest version of Windows SDK; if you're compiling the DX9 version, you need to download the June, 2010 version of DirectX 9 SDK (the DirectX 9 SDK can not be installed on Windows 8).

3. Import the FrameWork, GameTheory, Graph, ScriptEngine, and the GraphPlayer project into the same solution space.

4. To set up the compiling environment: in Visual Studio 2012 or 2013, right click on the project and select "Properties". From the "Properties" menu, select the VC++ Directories under the Configuration Properties. Add your WindowsSDK or DX9SDK in the include directories and library directories respectively. (note: in a lower version of Visual studio, the VC++ Directories page is on Tools-Options-Projects and Solutions).

5. Set the project dependences and the start up project. The GraphPlayer project should be set as the start up project, and set the project dependences of the GraphPlayer project to depends on all other projects (make sure other projects are build as .lib (standard library) under the general property page.

A.3 User Manual

A.3.1 The Main Scene

Open the Graph Simulator, you will see a user interface depicted as Figure A.1

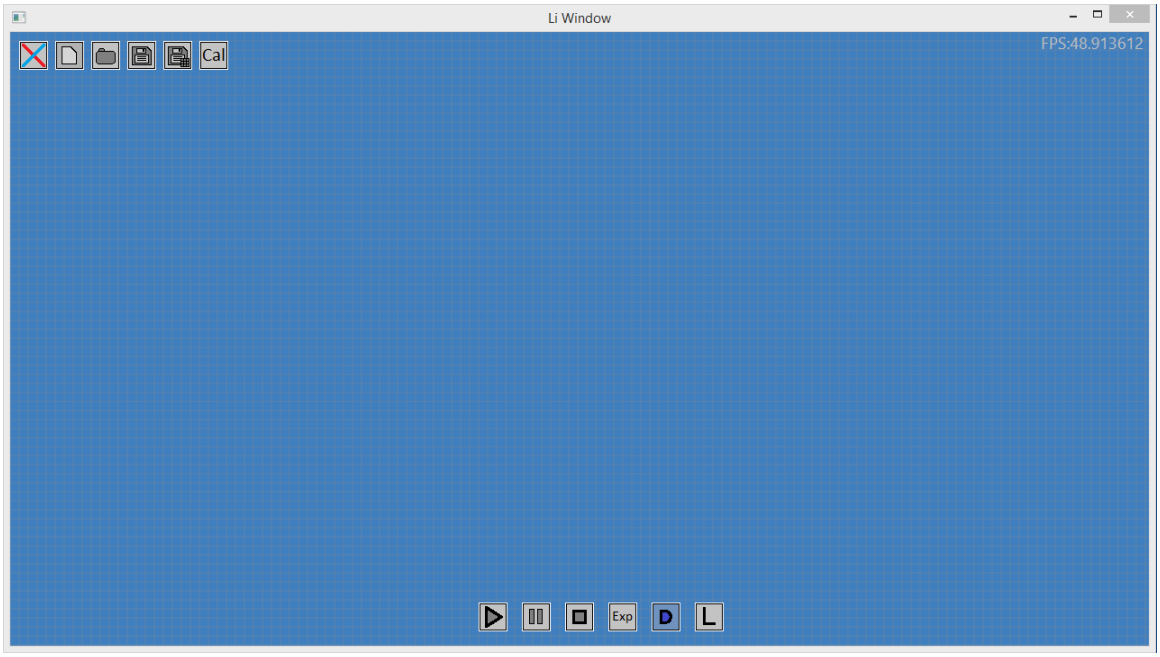

Fig. A.1 The Graph Player main interface

The entire window is the graph browser. Once you open a graph, the graph will be displayed on this window. There are two tool-bars on the top-left and centre-bottom of the screen, the one on the top is the graph control bar, and the other one is the game control bar.

A.3.2 The Graph Control Toolbar

The graph control toolbar is on the top left corner of the main interface (Figure A.2).

Fig. A.2 The Graph Control Bar

The Graph Display Layout List

: The Graph Display Layout List is a list which is used to set the graph's display. If you move your mouse on to the layout list, it will expend as follows: . You can select layout from the provided list; this will only change the display layout that the graph have been displayed on the screen, the graph topology remains the same.

- is the default layout. Under this display layout, the user is allowed to change any vertex's position in the graph at any time.
- is the ring layout. It will display the graph as a circle if the graph has less than 200 vertices (see Figure A.3), and as a spiral the graph with 200 nodes or more, with 200 nodes per period (see Figure A.4). This display layout is suit for displaying graphs with small world attributes.
- is the square layout. This layout will fit the vertices into a square (see Figure A.5), which is generally used to display regular graphs especially the lattice grid.
- is a layout that rank the nodes by their degrees. The higher the degree, the closer the node will be to the top-left corner of the browser.

New Document

: Click this button to start a new document, which will empty everything in the current scene. (Please make sure to save the graph you created before clicking this button.)

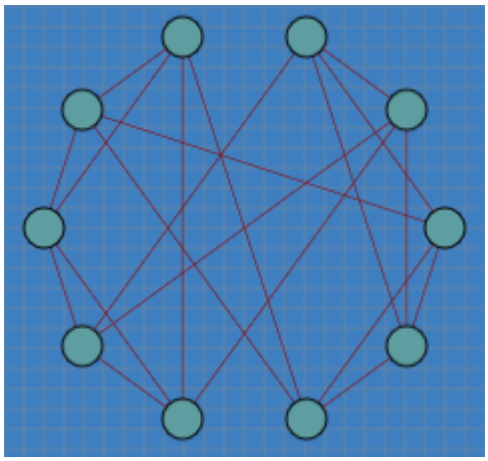

Fig. A.3 Graph displayed using ring layout

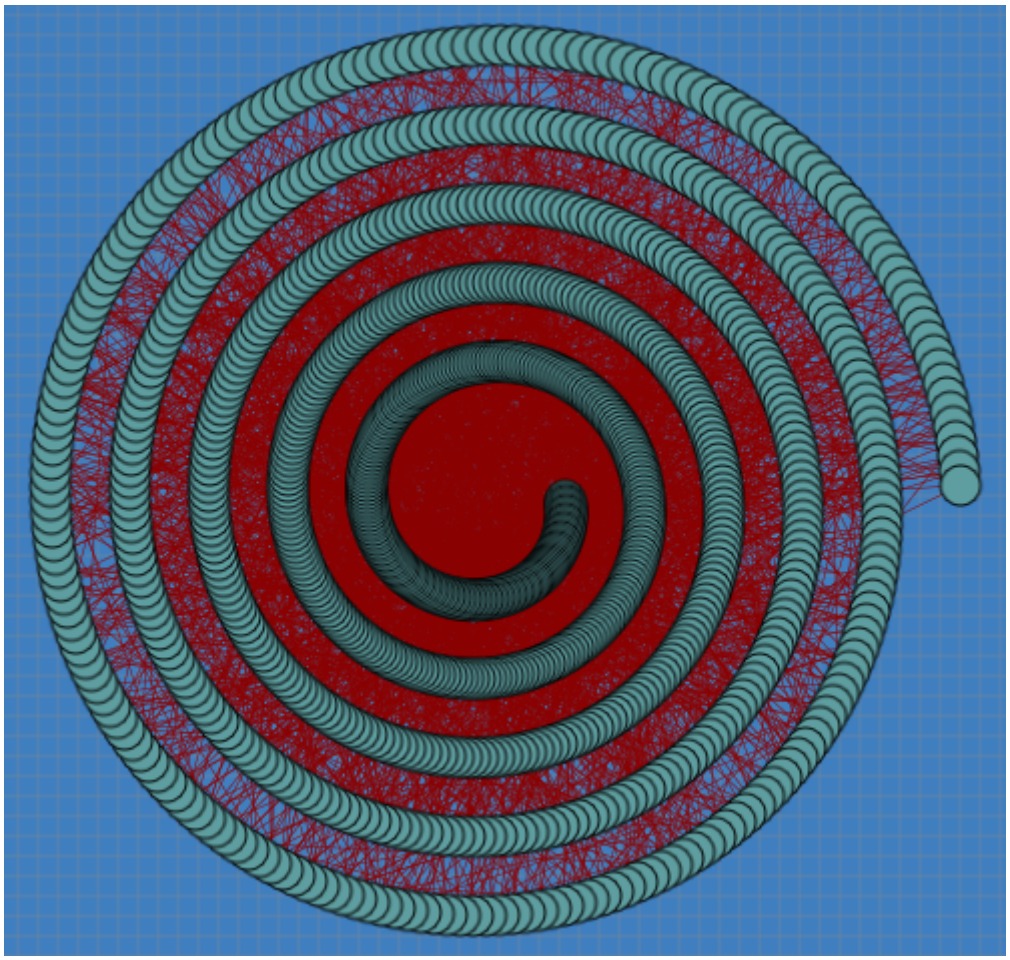

Fig. A.4 Graph displayed using the Spiral layout

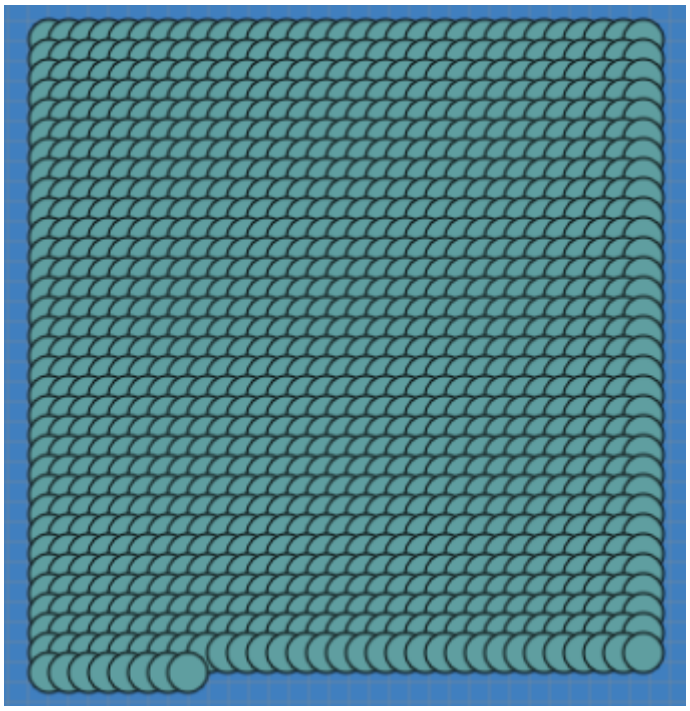

Fig. A.5 Graph displayed using the Square layout

Open an Existing Graph from File

: To open a graph from a file, you should click this button. Once you click the button, a open file window will be shown to let you select the file you want to open. (Please make sure the file you open is a valid graph file for the browser, which is the file exported from the graph browser. You can also write a graph file yourself or using other graph generator, the file format will be introduced later.)

Save the Graph

: To save the graph in the current scene, you can click this button, and input the file name, or select the file you want to save over. Then the graph browser will save your graph in the V/E (Vertex/Edge) format, which will introduced in the section of “File format” in the later of this Chapter. (You could give the file any extension name, you could also open and edit the file with any text editor).

Save as Adjacency Matrix

: As there are no standard file format for graph browsers, one of the common approach to sharing the graphs between different graph browsers is using an adjacency matrix. Clicking this button will allowed you to save your graph's adjacency matrix (see Chapter 2.3) into a plain text file.

Calculate High Time Complexity Attributes

: The graph browser will automatically calculate most of the graph for attributes and centralities once the graph has been loaded. However, the current algorithm to calculating some attributes and centralities, such as the betweenness and closeness, may have much higher time complexity than others. We put the calculation of those centrality into this button to promote the performance with large graphs (graphs more than $10,000$ vertices).

A.3.3 The Game Control Tool-bar

The game control toolbar is on the middle bottom of the main scene (Figure A.6).

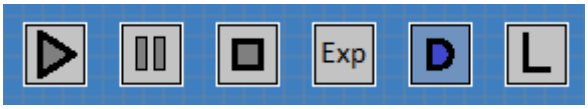

Fig. A.6 The Game Control Bar

Game Progress Control

One of the major tasks of the simulator is running the evolutionary games on the graph. In this simulator, we use the game control bar to control the progress of the evolution of the games, which includes :

- Click button (or press "1" on your keyboard) will start or continue the evolution.

- Click button (or press “2” on your keyboard) will pause the game in the middle of the evolution, which allows the user to check the state of the graph at any stage during the evolution.

- Click button (or press “3” on your keyboard) will stop the evolution and reset the population to the initial state.

Experiments Control

: Using the game progress control system, will run the experiment from the initial generation until the user stop it manually. If you need to run multiple experiments, and each of the experiment only been ran for a certain number of generations, you could use the experiment control button. The experiments control button will restart the evolution from the initial state when after 100 generations. Together with the log system, the simulator can run multiple experiments by itself.

Display the Graphic

: As many graphic applications, we have locked the maximum FPS (frames per second) at 60. However, normally, the evolution goes faster than that. To allow every generation in the evolution could be shown, we only update one generation per frame, which means the evolution may be slowed down if we display the graph graphically. This switch is used to turn the display on and off. When switched off, the graphics will no-longer be displayed, it allowed the evolution to run as fast as it can.

Graphics are switched on by default.

Log the Result

: When the log switch have been turned on, once you load a graph from a file, the detail of every experiment you have ran will be recorded in the corresponding log file. The log file’s name is setted as “yourfilename_log.txt” automatically.

This switch is switched off by default.

A.3.4 File Format

The simulator supports two file formats:

The Vertices/Edges (V/E) format

The Vertices / Edges (V/E) format is the default format of the simulator. By storing only the connection of each vertex, the V/E format has a maximum ($N \times N \times sizeof(int)$) storage cost (This storage costs is at its maximum value when the graph is a complete graph). For graphs with a lower average degree, the V/E format costs much lesser storage than the adjacency matrix format, which also means the graph browser will spend lesser time opening in comparison with the adjacency matrix format (the opening time may substantially lesser if the graph is sufficiently big).

Further more, we could also store some graph properties and centralities in the V/E file, which will save some time to calculate it every time we open the file. The graphic layout information for the browser can also be stored in the V/E format.

For a sample graph A.7 :

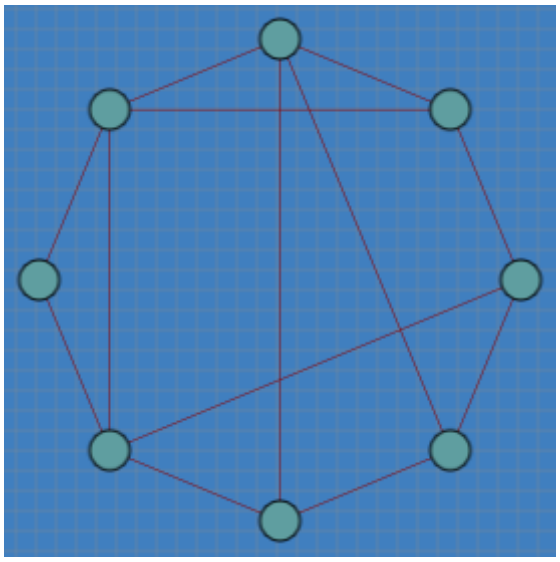

Fig. A.7 Sample Graph

The V/E format file of the graph A.7 is showed in table A.1

From table A.1, we can see, in the V/E format, the 8 on the second line is the number of vertices in the graph. The first column is the index of each node. The second and third column, are the coordinates for displaying in the graph browser, the fourth column is the de-

Li_Generate_Graph:

```
8
0   260       180       3   {   7   1   3   };
1   236.569   236.569   3   {   0   2   6   };
2   180       260       3   {   1   3   6   };
3   123.431   236.569   4   {   2   4   5   0   };
4   100       180       2   {   3   5   };
5   123.431   123.431   4   {   6   4   3   7   };
6   180       100       4   {   5   7   1   2   };
7   236.569   123.431   3   {   6   0   5   };
```

Table A.1 V/E file format sample

gree of the corresponding vertex. The values in the brackets are the indexes of all connected nodes.

The V/E format file can be opened with the open button . To save a graph to the V/E format, you can click on the save button .

Adjacency Matrix

Besides the default V/E format, the simulator can also store the graph's adjacency matrix in a plain text file, which have been introduced earlier in the Chapter 2.3. A sample adjacency matrix format file for graph A.7 is shown below in table A.2

Li_Generate_Graph:

```
8
{   0   1   0   1   0   0   0   1   };
{   1   0   1   0   0   0   1   0   };
{   0   1   0   1   0   0   1   0   };
{   1   0   1   0   1   1   0   0   };
{   0   0   0   1   0   1   0   0   };
{   0   0   0   1   1   0   1   1   };
{   0   1   1   0   0   1   0   1   };
{   1   0   0   0   0   1   1   0   };
```

Table A.2 Adjacency matrix file format sample

The adjacency format has been used by some other graph browsers. This graph browser currently do not support an adjacency matrix file, however, it can save the V/E graph to a

adjacency matrix file, in order to transfer graphs between different graph browsers. To save your graph's adjacency matrix, click the button.

A.3.5 Create and Browse Graphs Using the Graph Browser

Besides generating graphs with an extra graph generator, one can easily create a graph using the graphic interface visually. The browser supplies several graph operations for the users to generate graphs manually.

- Add a vertex: Left click anywhere (where there is no vertex) of the canvas while holding "Shift" button, will add a new vertex to the position.
- Remove a vertex: Put the cursor on the vertex and press "Delete" button will remove the corresponding vertex.
- Add an edge: To add an edge between two vertex, hold "Ctrl", then click on the two vertices that you want to connect.
- Remove an edge: To add an edge between two vertex, hold "Tab", then click on the two vertices (which already connected) that you want to disconnect successively.
- Move the canvas: To move the entire canvas, holding the right button and move the mouse.
- Scale the canvas: To scale the canvas, use the mouse wheel.
- Display the properties of the graph: Once a graph have been opened correctly, the graph properties will be automatically displayed on the top-right of the interface (figure A.8).

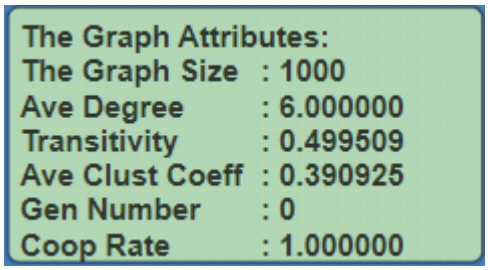
The Graph Attributes:
The Graph Size : 1000
Ave Degree : 6.000000
Transitivity : 0.499509
Ave Clust Coeff : 0.390925
Gen Number : 0
Coop Rate : 1.000000

Fig. A.8 Sample of the Graph Properties

- Display the centralities of a vertex : Move your mouse on the vertex, the centralities will be displayed on the right hand side of your mouse. The vertex with your cursor on it will be surrounded by a white circle.
- Move a vertex: Left click and hold on an vertex, and move the mouse will allow you to move the position of the corresponding vertex. This will only work when using the free graph display layout . The vertex that has been selected will be surrounded by a green circle.
- Select multiple vertices: Click and hold your left mouse button on the canvas, then move your mouse. This will draw a black rectangle on the canvas. The vertices inside the rectangle will be surrounded by green circles. Releasing the left mouse button will select all the vertices in green circles (figure A.9).

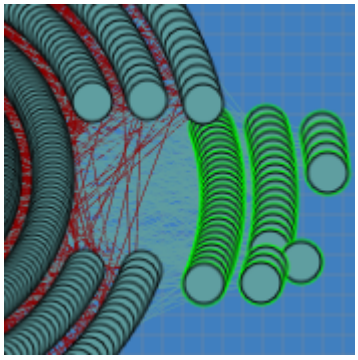

Fig. A.9 Sample of Multi-Selection

- Move multiple vertices: Once you have selected multiple vertices, you can move them together by clicking and holding your left mouse button on anywhere of the canvas and then moving your mouse.
- Remove multiple vertices: Once you selected multiple vertices, you can remove all of them by pressing the “delete” button.
- Make a clique: To make a clique (complete sub-graph), select the vertices you need and press “S”.
- Display all neighbours : move your mouse over a vertex, and press “G”. All neighbours of that vertex will be gathered together and surround that vertex in a circle A.10.

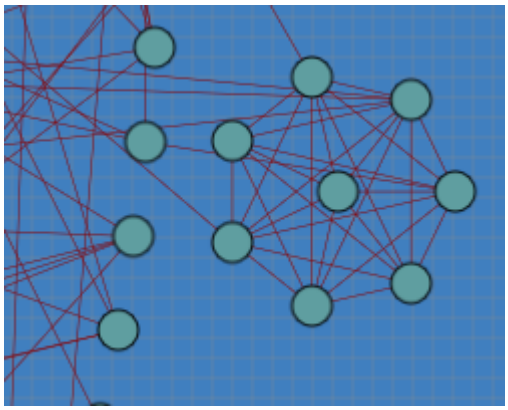

Fig. A.10 Sample of displaying all neighbours for a node

A.3.6 Setting up the Games

Once we open a graph, all of the vertices and edges will be displayed by default (if the graph has more than 5000 edges, the edges will be hidden for the performance issues). All blue vertices will adopt the "cooperate" strategy (or C for short) in the evolution. By double clicking on the vertex, (or moving your mouse on the vertex and press "D"), you can change the vertex's strategy to "Defect" (or D for short). Once the individual is defecting, the corresponding vertex in the graph will be displayed as a red node. You can also double click the node (or move your mouse on the vertex and press "C") to set it back to cooperate.

In the "GameSetting.ini" file, you could change the payoff matrix of the game that will be played in the simulator.

A.3.7 Using the LUA Console to Scripting

The simulator can also supports LUA scripts. You can use LUA scripts to generate your graphs as well. The scripts can be read from either a console in the simulator or through script files.

To open the LUA console, you need press the " $\sim$ " button on your keyboard (normally on the left of "1"). A console will be open from the top of the scene (figure A.11).

There are currently two libraries of LUA functions have been created for the simulator, the "gamelib" and "graphlib".

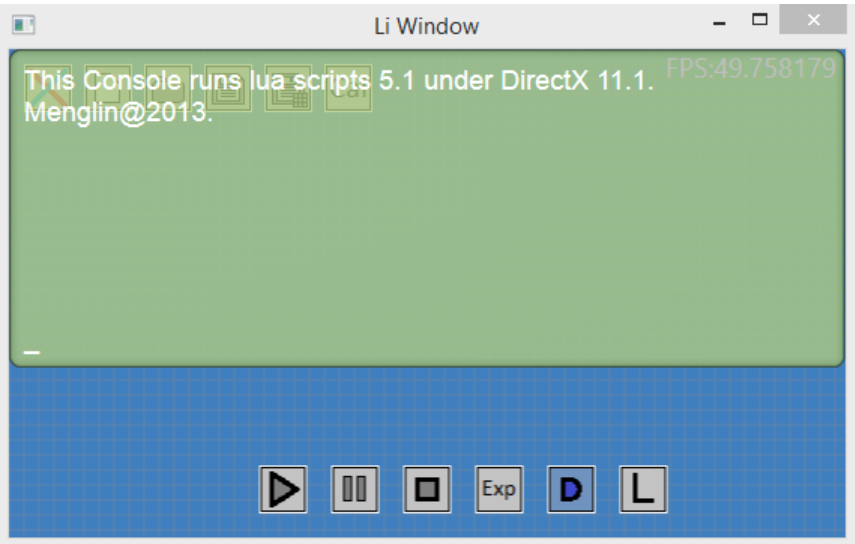

Fig. A.11 Opening the Lua Console

The "gamelib"

The game library is used to control the game's payoff without reloading the "gamesetting.ini" file. Currently, you can use it to display or change the value of the temptation payoff of the game, by inputting "gamelib.getT();", or "gamelib.setT(t);" in your console A.12.

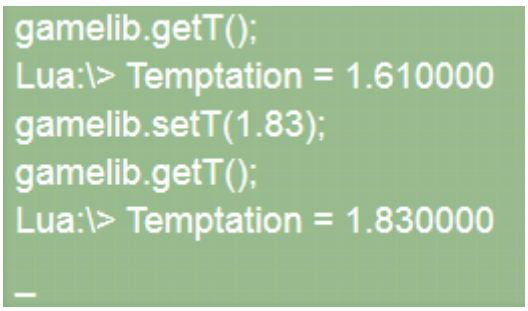

Fig. A.12 Using Lua function to get and set the temptation payoff

The "graphlib"

The graph library can be used as an alternative approach to generate and / or modify graphs. Several functions have been provided as follow :

- "addNode" : By inputting "graphlib.addNode();" in the console, you can add a vertex

in the graph A.13. it can also be used as “graphlib.addNode(x, y);” to identify the x and y position of the vertex. If you did not specify a position, the vertex will be automatically added at $(0,0)$.

- “addEdge” : By inputting “graphlib.addEdge(a,b);” in the console, you can connect vertex a and b with an edge (figure A.13).

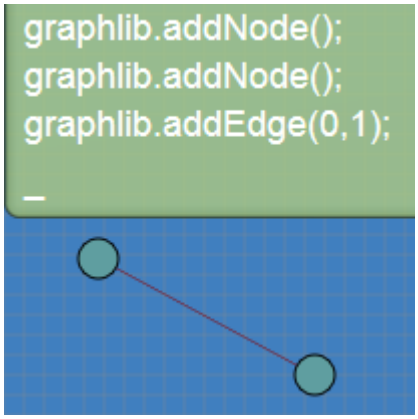

Fig. A.13 Using Lua functions to add Nodes and Edges

- “removeNode” : By inputting “graphlib.removeNode(i);” in the console, the node with index “i” will be removed from the graph with all its connections.
- “clearGraph” : Input “graphlib.clearGraph();” to remove all nodes and edges on the current canvas.

Together with the above functions, you could use the lUA script to create larger graphs in the simulator graphically which would be time consuming to create by mouse click (Figure A.14).

Also, you can write the scripts into a file, and load the file with “dofile(“filename.lua”)” in the console. For example, if we want to create an $N = 2$ small world graph with 100 vertices, we could do that in LUA script by creating a “sw.lua” file, and type in:

```
for i=0, 99 do
    graphlib.addNode(i*10, i*10);

    if i == 0 then

    elseif i == 1 then
```

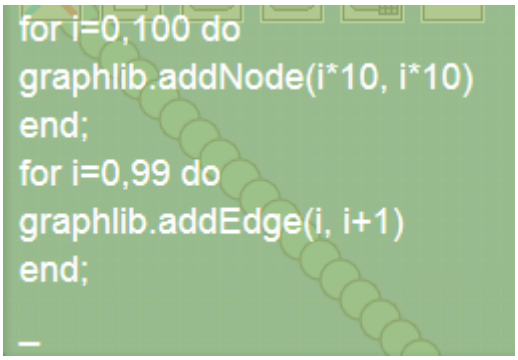

Fig. A.14 More lua scripts in console.

```
        graphlib.addEdge(0,1);
    else
        graphlib.addEdge(i-1, i);
        graphlib.addEdge(i-2, i);
    end;

    if i == 98 then
        graphlib.addEdge(0, 98);
    elseif i == 99 then
        graphlib.addEdge(0, 99);
        graphlib.addEdge(1, 99);
    end;
end;
```

In the simulator, by simply typing "dofile("sw.lua")", a $size = 100$, $N = 2$ small world graph will be generated.

A.4 HotKey List

- "shift" : Add a node;
- "ctrl" : Add an edge;
- "delete" : Remove a node;
- "alt" : Remove an edge;

- “1” : Run the evolution;
- “2” : Pause the evolution;
- “3” : Stop the evolution;
- “C” : Set the strategy of a selected node to cooperate;
- “D” : Set the strategy of a selected node to defect;
- “S” : Build a clique (Complete sub graph) for selected nodes;
- “G” : Display all neighbours of a selected node.
- “R” : Reinitialise the graph (set all strategies and display layout back to the initialise setting).
- “M” : Merge all cliques into one node.

A.5 summary

The simulator gives us a fast and reliable way to run the evolutionary game’s simulation using computer. It can also provide us a graphic interface to the data, and a friendly user interface to access the data. Basic log / analysis tools also provide by the simulator, which are very handy to the user to record their experiment’s result.

Appendix B

Publications

1. Menglin Li, Colm O'Riordan. **Graph Centrality Measures and the robustness of cooperation.** *CEC: Congress on Evolutionary Computation 2014.* June, Beijing, China

2. Menglin Li, Colm O'Riordan. **Analysis of generalised tit-for-tat strategies in evolutionary spatial N-player prisoner dilemmas.** *GECCO: Genetic and Evolutionary Computation Conference 2013.* July 06-10, Amsterdam, Neitherland

3. Menglin Li, Colm O'Riordan. **The effect of clustering coefficient and node degree on the robustness of cooperation.** *CEC: Congress on Evolutionary Computation.* 2013 June 20-23, Cancun, Mexico